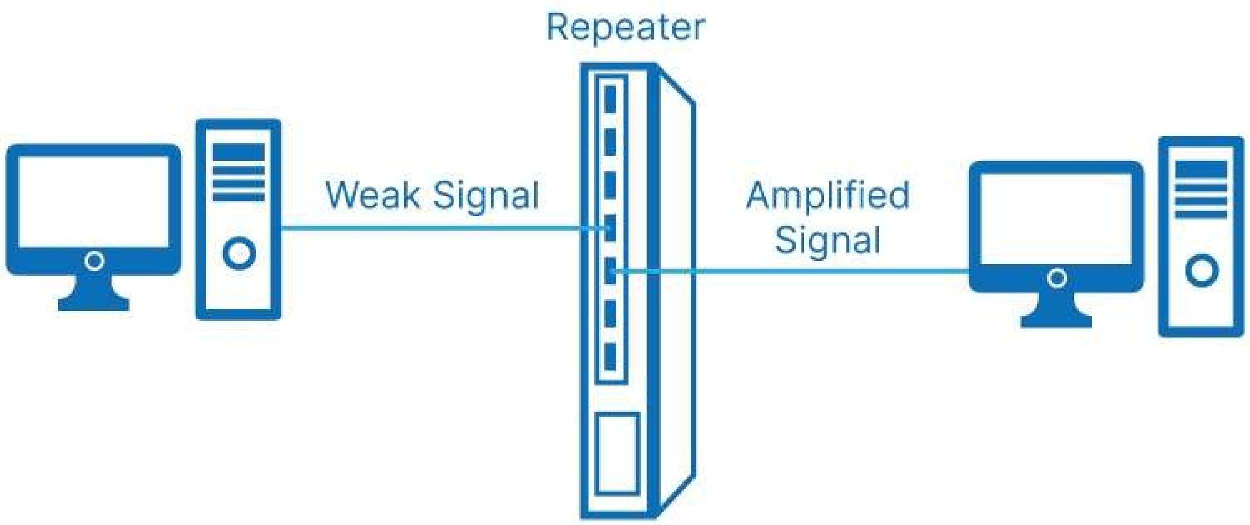
Exercise:1

Study of Network devices in detail and connect the computers in Local Area Network.

Network Devices: Network devices, also known as networking hardware, are physical devices that allow hardware on a computer network to communicate and interact with one another. For example Repeater, Hub, Bridge, Switch, Routers, Gateway, Brouter, and NIC, etc.

1. Repeater *– A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they not only amplify the signal but also regenerate it. When the signal becomes weak, they copy it bit by bit and regenerate it at its star topology connectors connecting following the original strength. It is a 2-port device.*



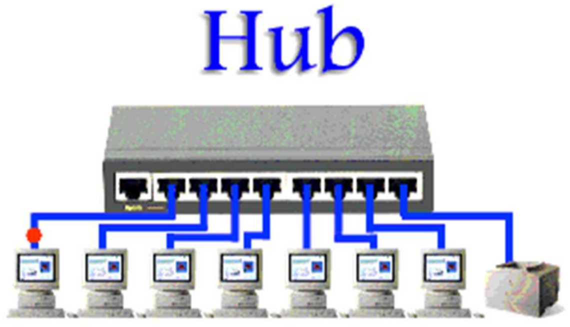
2. Hub – *A hub is a basically multi-port repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices. In other words, the collision domain of all hosts connected through Hub remains one. Also, they do not have the intelligence to find out the best path for data packets which leads to inefficiencies and wastage.*

Types of Hub

Active Hub:- *These are the hubs that have their power supply and can clean, boost, and relay the signal along with the network. It serves both as a repeater as well as a wiring center. These are used to extend the maximum distance between nodes.*

Passive Hub:- *These are the hubs that collect wiring from nodes and power supply from the active hub. These hubs relay signals onto the network without cleaning and boosting them and can't be used to extend the distance between nodes.*

Intelligent Hub:- *It works like an active hub and includes remote management capabilities. They also provide flexible data rates to network devices. It also enables an administrator to monitor the traffic passing through the hub and to configure each port in the hub.*

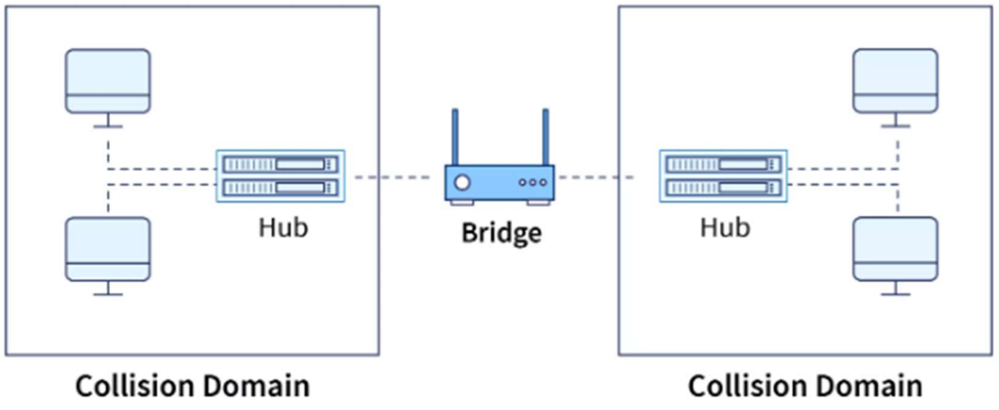


3. Bridge – *A bridge operates at the data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of the source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2 port device.*

Types of Bridges

Transparent Bridges:- *These are the bridge in which the stations are completely unaware of the bridge's existence i.e. whether or not a bridge is added or deleted from the network, reconfiguration of the stations is unnecessary. These bridges make use of two processes i.e. bridge forwarding and bridge learning.*

Source Routing Bridges:- *In these bridges, routing operation is performed by the source station and the frame specifies which route to follow. The host can discover the frame by sending a special frame called the discovery frame, which spreads through the entire network using all possible paths to the destination.*



4. Switch – *A switch is a multiport bridge with a buffer and a design that can boost its efficiency(a large number of ports imply less traffic) and performance. A switch is a data link layer device. The switch can perform error checking before forwarding data, which makes it very efficient as it does not forward packets that have errors and forward good packets selectively to the correct port only. In other words, the switch divides the collision domain of hosts, but the broadcast domain remains the same.*

Types of Switch

Unmanaged switches*: These switches have a simple plug-and-play design and do not offer advanced configuration options. They are suitable for small networks or for use as an expansion to a larger network.*

Managed switches: *These switches offer advanced configuration options such as VLANs, QoS, and link aggregation. They are suitable for larger, more complex networks and allow for centralized management.*

Smart switches: *These switches have features similar to managed switches but are typically easier to set up and manage. They are suitable for small- to medium-sized networks.*

Layer 2 switches: *These switches operate at the Data Link layer of the OSI model and are responsible for forwarding data between devices on the same network segment.*

Layer 3 switches: *These switches operate at the Network layer of the OSI model and can route data between different network segments. They are more advanced than Layer 2 switches and are often used in larger, more complex networks.*

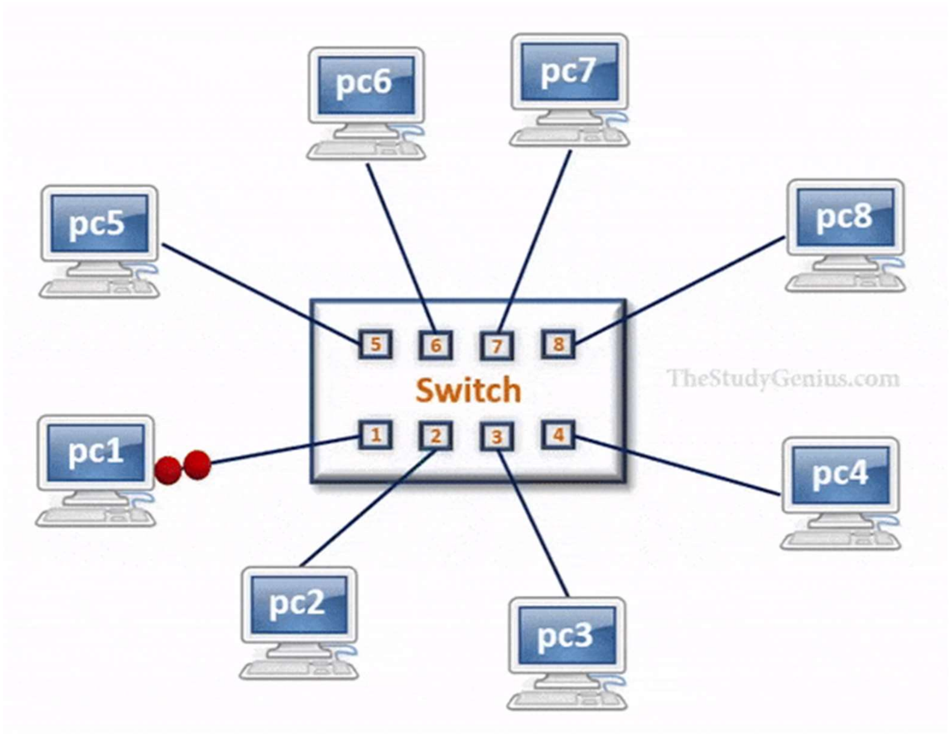
PoE switches: *These switches have Power over Ethernet capabilities, which allows them to supply power to network devices over the same cable that carries data.*

Gigabit switches: *These switches support Gigabit Ethernet speeds, which are faster than traditional Ethernet speeds.*

Rack-mounted switches: *These switches are designed to be mounted in a server rack and are suitable for use in data centers or other large networks.*

Desktop switches: *These switches are designed for use on a desktop or in a small office environment and are typically smaller in size than rack-mounted switches.*

Modular switches: *These switches have modular design, which allows for easy expansion or customization. They are suitable for large networks and data centers.*



5. Routers – *A router is a device like a switch that routes data packets based on their IP addresses. The router is mainly a Network Layer device. Routers normally connect LANs and WANs and have a dynamically updating routing table based on which they make decisions on routing the data packets. The router divides the broadcast domains of hosts connected through it.*

6. Gateway – *A gateway, as the name suggests, is a passage to connect two networks that may work upon different networking models. They work as messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switches or routers. A gateway is also called a protocol converter.*

7. Brouter – *It is also known as the bridging router is a device that combines features of both bridge and router. It can work either at the data link layer or a network layer. Working as a router, it is capable of routing packets across networks and working as the bridge, it is capable of filtering local area network traffic.*

8. NIC – *NIC or network interface card is a network adapter that is used to connect the computer to the network. It is installed in the computer to establish a LAN. It has a unique id that is written on the chip, and it has a connector to connect the cable to it. The cable acts as an interface between the computer and the router or modem. NIC card is a layer 2 device which means that it works on both the physical and data link layers of the network model.*

Exercise:2

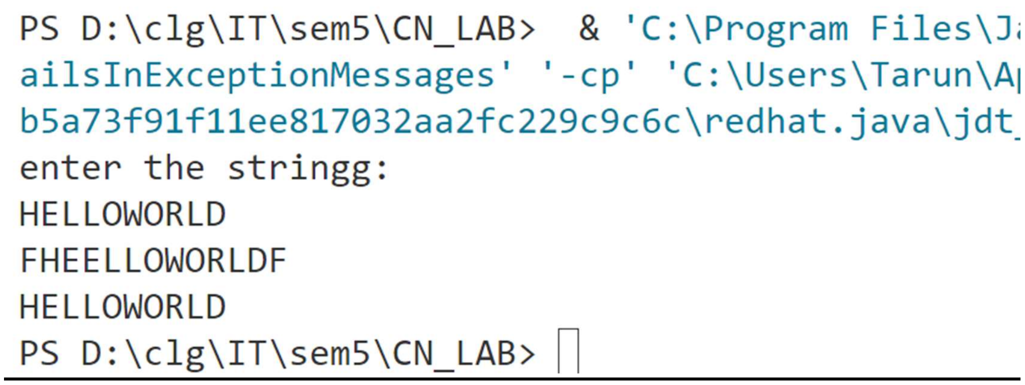
Write a Program to implement the data link layer farming methods such as
i) Character stuffing ii) bit stuffing.

i) Character stuffing

program:

```
import java.util.Scanner;
import java.util.*;
public class bytestuff {
   public static void main(String[] args) {
      Scanner sc=new Scanner(System.in);
      System.out.println("enter the stringg:");
      String x=sc.next();
      String op="F";
      for (int i = 0; i < x.length(); i++)
      {
         if (x.charAt(i)=='E') {op=op+"EE";}
         else if (x.charAt(i)=='F') {op=op+"EF";}
         else op=op+x.charAt(i);
      }
      op=op+'F';
      System.out.println(op);
     String d="";
      op=op.replaceFirst("F", "");
      op=op.replace("EE", "E");
      op=op.replace("EF", "F");
      for (int i = 0; i < op.length()-1; i++)
      {
         d=d+op.charAt(i);
      }
      System.out.println(d);    }}
```

Output:



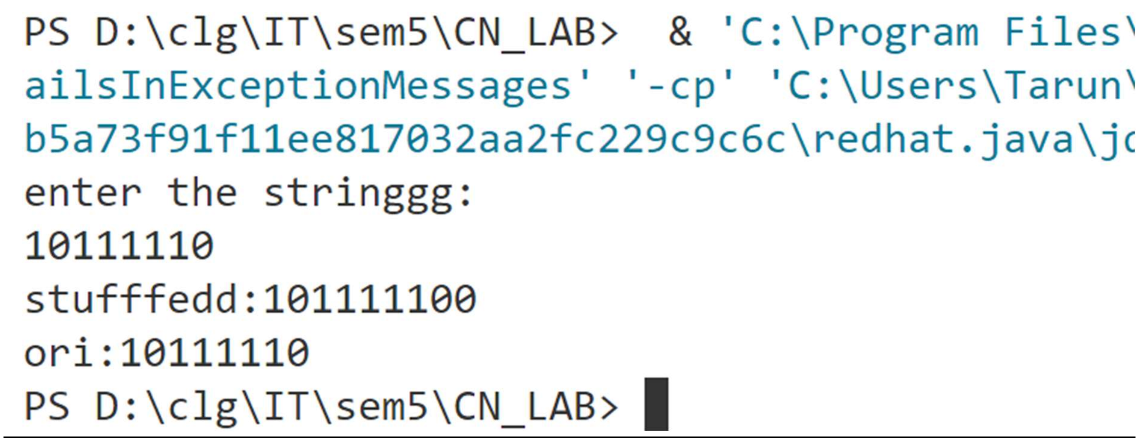
```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Files\Ja
ailsInExceptionMessages' '-cp' 'C:\Users\Tarun\Ap
b5a73f91f11ee817032aa2fc229c9c6c\redhat.java\jdt_
enter the stringg:
HELLOWORLD
FHEELLOWORLDF
HELLOWORLD
PS D:\clg\IT\sem5\CN_LAB>
```

ii) bit stuffing.

<u>Program:</u>

```
import java.util.Scanner;
public class bitstuff
{
    public static void main(String[] args) {
        Scanner sc =new Scanner(System.in);
        System.out.println("enter the stringgg:");
        String str=sc.next();
        String op="";
        int cnt=0;
        for(int i =0; i<str.length();i++)
        {
            if (str.charAt(i)=='1')
            { cnt++;
            }
            else cnt=0;
            op=op+str.charAt(i);
            if(cnt==5)
            {
                cnt=0;
                op=op+"0";
            }
        }
        System.out.println("stufffedd:" + op);
        int cntt=0;
        String opp="";
        for(int i =0; i<str.length();i++)
        {
            if (str.charAt(i)=='0')
            { cntt++;
            }
            else cntt=0;
            opp=opp+str.charAt(i);
            if(cntt==5)
            {
              cntt=0;
              opp=opp;
            }
          }
          System.out.println("ori:" + opp);
        }
     }
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Files\
ailsInExceptionMessages' '-cp' 'C:\Users\Tarun\
b5a73f91f11ee817032aa2fc229c9c6c\redhat.java\jd
enter the stringgg:
10111110
stufffedd:101111100
ori:10111110
PS D:\clg\IT\sem5\CN_LAB>
```

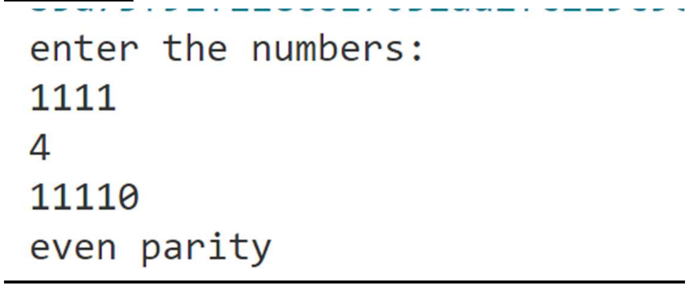
Exercise:3

Write a Program to implement data link layer farming method checksum.

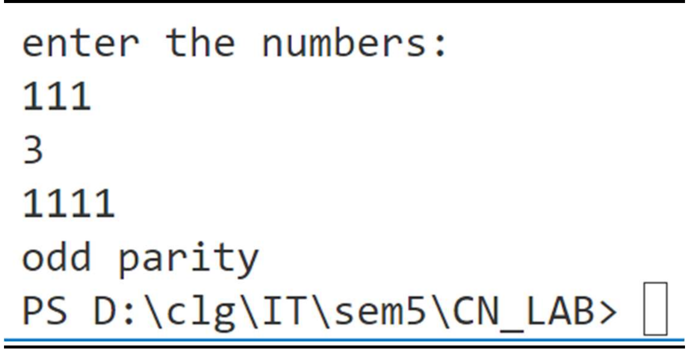
Program:

```
import java.util.Scanner;
public class chksum
{
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        System.out.println("enter the numbers:");
        String x=sc.next();
        String y=x;
        int c=0,par=0;
        for (int i = 0; i < x.length(); i++)
        {
            if(x.charAt(i)=='1' )
            {
                c++;
            }

        }
        System.out.println(c);
        if(c%2==0)
        {
            c=0;
            par=0;
            y=y+"0";
            System.out.println(y);

        }
        else {c=0;y=y+1; System.out.println(y); par=1;}
        for (int i = 0; i < y.length()-1; i++)
        {
           if(y.charAt(i)=='1')
           {
              c++;
           }
        }
           if(c%2==0)
           {
              if(par==0){System.out.println("even parity");}
           }
           if(c%2!=0)
           {
              if(par==1){System.out.println("odd parity");}
           }

    }

}
```

Output:



```
enter the numbers:
1111
4
11110
even parity
```



```
enter the numbers:
111
3
1111
odd parity
PS D:\clg\IT\sem5\CN_LAB>
```

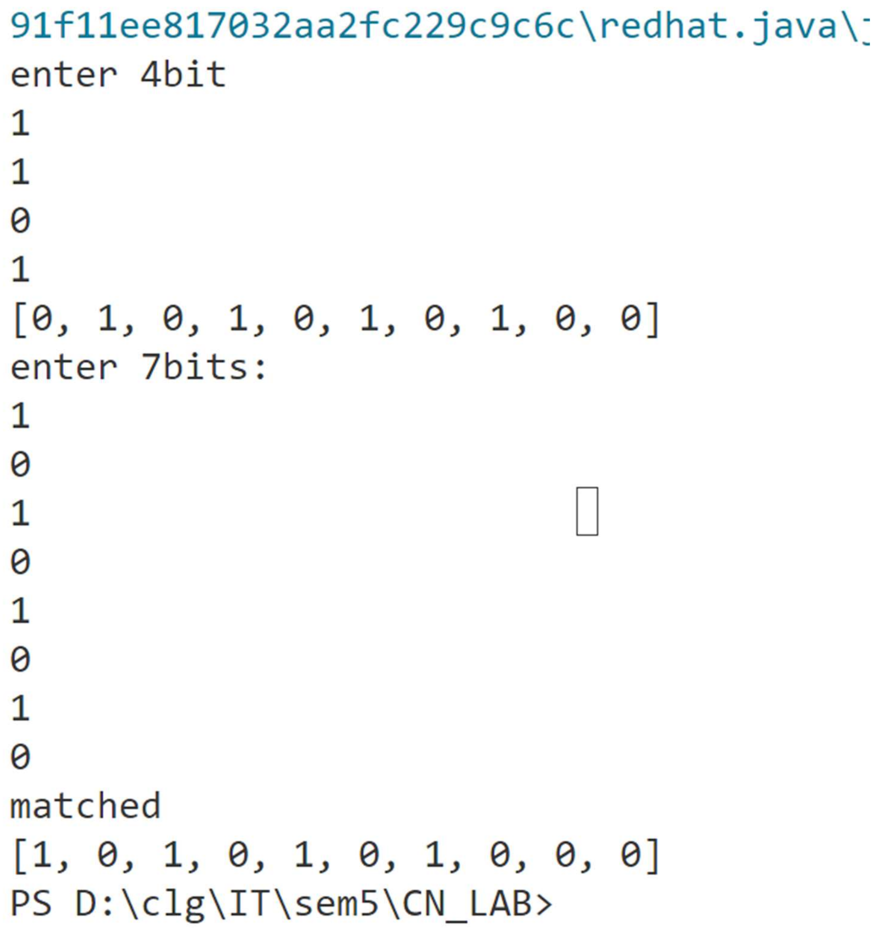
Exercise:4

Write a program for Hamming Code generation for error detection and correction.

Program:

```
import java.util.Arrays;
import java.util.Scanner;
public class hamming
{
    public static void main(String[] args) {
        int Arr[]=new int[10];
        Scanner sc=new Scanner(System.in);
          System.out.println("enter4bit");
        Arr[3]=sc.nextInt();
        Arr[5]=sc.nextInt();
        Arr[6]=sc.nextInt();
        Arr[7]=sc.nextInt();
        Arr[1]=Arr[3]^Arr[5]^Arr[7];
        Arr[2]=Arr[3]^Arr[6]^Arr[7];
        Arr[4]=Arr[5]^Arr[6]^Arr[7];
        System.out.println(Arrays.toString(Arr));
        int b[]=new int[10];System.out.println("enter 7bits:");
        for (int i = 0; i < 8; i++) {
            b[i]=sc.nextInt();
        }
        int c1=b[1]^b[3]^b[5]^b[7];
        int c2=b[2]^b[3]^b[6]^b[7];
        int c3=b[4]^b[5]^b[6]^b[7];
        int p=c1*1+c2*2+c3*4;
        if (p==0)
        {
            System.out.println("matched");
            System.out.println(Arrays.toString(b));
        }
        else{
           System.out.println("error at pos"  + p);
           if (b[p]==0)
           {
              b[p]=1;
           }
           else {b[p]=0;}
                    System.out.println(Arrays.toString(b));
        }
     }
}
```

Output:



```
91f11ee817032aa2fc229c9c6c\redhat.java\
enter 4bit
1
1
0
1
[0, 1, 0, 1, 0, 1, 0, 1, 0, 0]
enter 7bits:
1
0
1
0
1
0
1
0
matched
[1, 0, 1, 0, 1, 0, 1, 0, 0, 0]
PS D:\clg\IT\sem5\CN_LAB>
```

Exercise:5

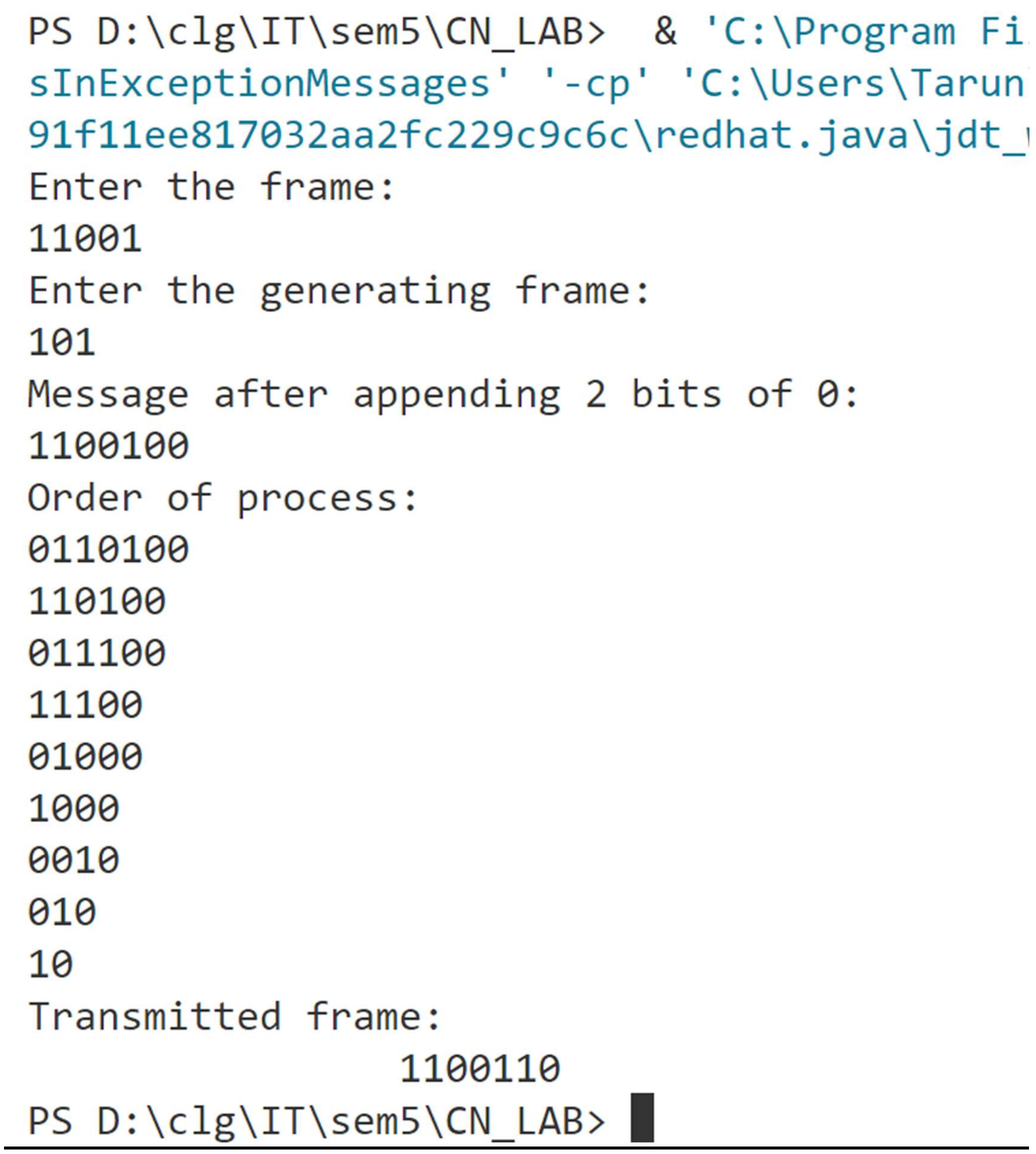
Write a Program to implement on a data set of characters the three CRC polynomials –CRC 12, CRC 16 and CRC CCIP.

Program:

```
import java.util.Scanner;
public class CRC {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the frame:");
        String s = scanner.nextLine();
        System.out.println("Enter the generating frame:");
        String d = scanner.nextLine();
        int k = d.length() - 1;
        int len = s.length();
        StringBuilder p = new StringBuilder(s);
        for (int i = len; i < len + k; i++) {
            p.append('0');
        }
        System.out.println("Message after appending " + k + " bits of
0:");
        System.out.println(p.toString());
        System.out.println("Order of process:");
        while (p.length() > k) {
            if (p.charAt(0) == '1') {
                for (int i = 0; i < d.length(); i++) {
                    int pBit = p.charAt(i) - '0';
                    int dBit = d.charAt(i) - '0';
                    p.setCharAt(i, (char) (pBit ^ dBit + '0'));
                }
            } else {
                p.deleteCharAt(0);
            }
```

```
          System.out.println(p.toString());
        }
        s += p.toString();
        System.out.println("Transmitted frame:");
        System.out.println("\t\t" + s);
    }
}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Fi
sInExceptionMessages' '-cp' 'C:\Users\Tarun
91f11ee817032aa2fc229c9c6c\redhat.java\jdt_
Enter the frame:
11001
Enter the generating frame:
101
Message after appending 2 bits of 0:
1100100
Order of process:
0110100
110100
011100
11100
01000
1000
0010
010
10
Transmitted frame:
                1100110
PS D:\clg\IT\sem5\CN_LAB> █
```

Exercise:6

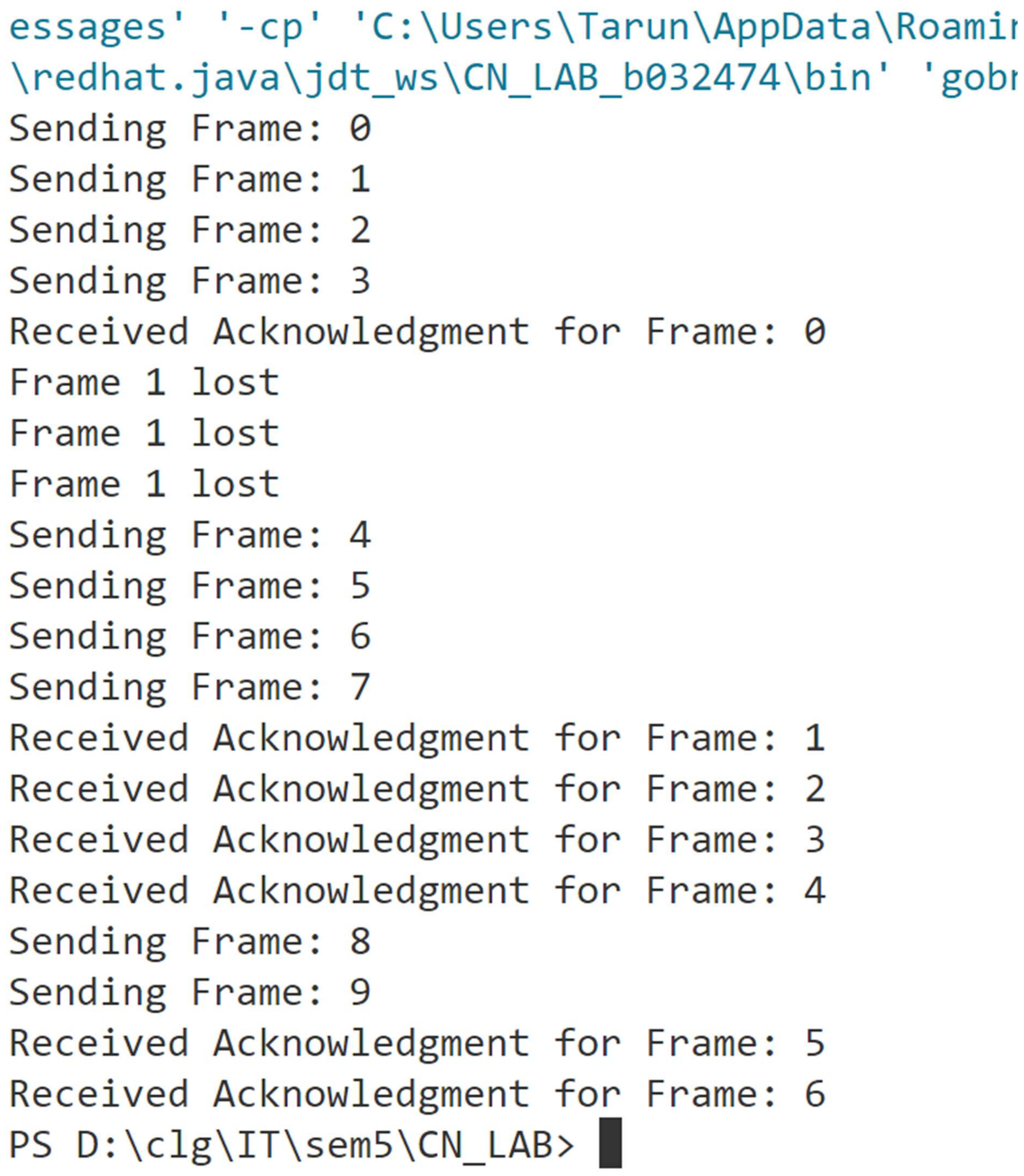
Write a Program to implement Sliding window protocol for Goback N.

Program:

```
import java.util.ArrayDeque;
import java.util.Queue;
import java.util.Random;
public class gobn {
   public static void main(String[] args) {
      int windowSize = 4;
      int totalFrames = 10;
      Queue<Integer> frameQueue = new ArrayDeque<>();
      for (int i = 0; i < totalFrames; i++) {
         frameQueue.add(i);
      }
      Random random = new Random();
      int expectedAck = 0;

      while (!frameQueue.isEmpty()) {
         int framesInTransit = 0;
         while(framesInTransit<windowSize&&  !frameQueue.isEmpty())
{int frame = frameQueue.poll();
            System.out.println("Sending Frame: " + frame);
            framesInTransit++;
         }
         for (int i = 0; i < framesInTransit; i++) {
            if (random.nextDouble() < 0.2) {
               System.out.println("Frame " + expectedAck + " lost");
            } else {
System.out.println("Received   Acknowledgment   for   Frame:   "   +
expectedAck);
               expectedAck++;
            }
         }     }   } }
```

Date: ____/____/2023

Output:



```
essages' '-cp' 'C:\Users\Tarun\AppData\Roami
\redhat.java\jdt_ws\CN_LAB_b032474\bin' 'gobr
Sending Frame: 0
Sending Frame: 1
Sending Frame: 2
Sending Frame: 3
Received Acknowledgment for Frame: 0
Frame 1 lost
Frame 1 lost
Frame 1 lost
Sending Frame: 4
Sending Frame: 5
Sending Frame: 6
Sending Frame: 7
Received Acknowledgment for Frame: 1
Received Acknowledgment for Frame: 2
Received Acknowledgment for Frame: 3
Received Acknowledgment for Frame: 4
Sending Frame: 8
Sending Frame: 9
Received Acknowledgment for Frame: 5
Received Acknowledgment for Frame: 6
PS D:\clg\IT\sem5\CN_LAB>
```

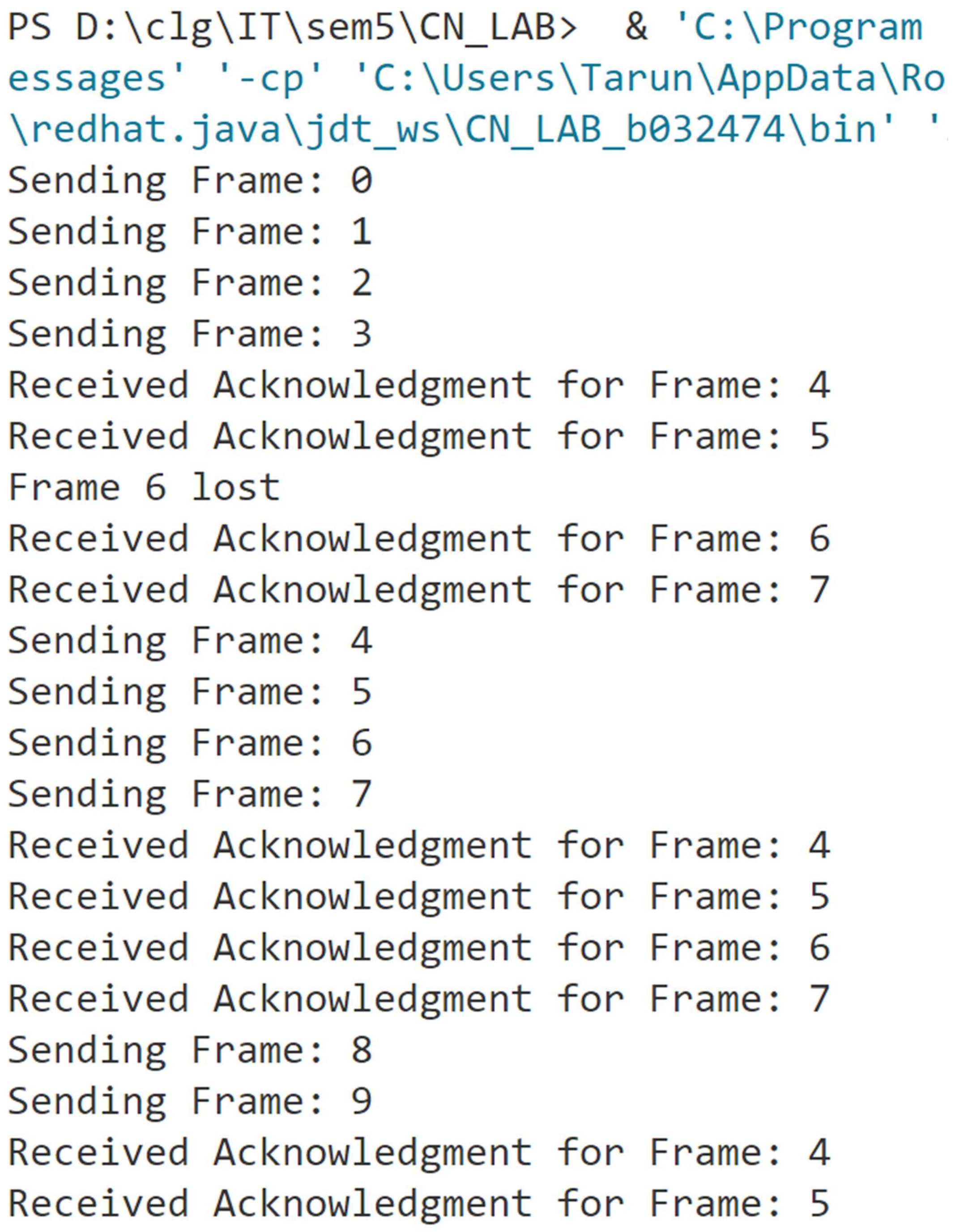
Exercise:7

Write a Program to implement Sliding window protocol for Selective repeat.

Program:

```
import java.util.ArrayDeque;
import java.util.Queue;
import java.util.Random;
public class SelectiveRepeatSender {
    public static void main(String[] args) {
        int windowSize = 4; // Window size
        int totalFrames = 10; // Total number of frames to be sent
        Queue<Integer> frameQueue = new ArrayDeque<>();
        for (int i = 0; i < totalFrames; i++) {
            frameQueue.add(i);
        }
        Random random = new Random();
        while (!frameQueue.isEmpty()) {
            int framesInTransit = 0;
            int frameBeingAcknowledged = 0;
            while        (framesInTransit        <        windowSize        &&
!frameQueue.isEmpty()) {
                int frame = frameQueue.poll();
                System.out.println("Sending Frame: " + frame);
                framesInTransit++;
            }
            while (frameBeingAcknowledged < framesInTransit) {
                if (random.nextDouble() < 0.2) { // Simulate frame loss
                    System.out.println("Frame " + (frameBeingAcknowledged +
windowSize) + " lost");
                } else {
                    System.out.println("Received Acknowledgment for Frame:
" + (frameBeingAcknowledged + windowSize));
                    frameBeingAcknowledged++;
                }          }       }    }}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program
essages' '-cp' 'C:\Users\Tarun\AppData\Ro
\redhat.java\jdt_ws\CN_LAB_b032474\bin' '
Sending Frame: 0
Sending Frame: 1
Sending Frame: 2
Sending Frame: 3
Received Acknowledgment for Frame: 4
Received Acknowledgment for Frame: 5
Frame 6 lost
Received Acknowledgment for Frame: 6
Received Acknowledgment for Frame: 7
Sending Frame: 4
Sending Frame: 5
Sending Frame: 6
Sending Frame: 7
Received Acknowledgment for Frame: 4
Received Acknowledgment for Frame: 5
Received Acknowledgment for Frame: 6
Received Acknowledgment for Frame: 7
Sending Frame: 8
Sending Frame: 9
Received Acknowledgment for Frame: 4
Received Acknowledgment for Frame: 5
```

Exercise:8

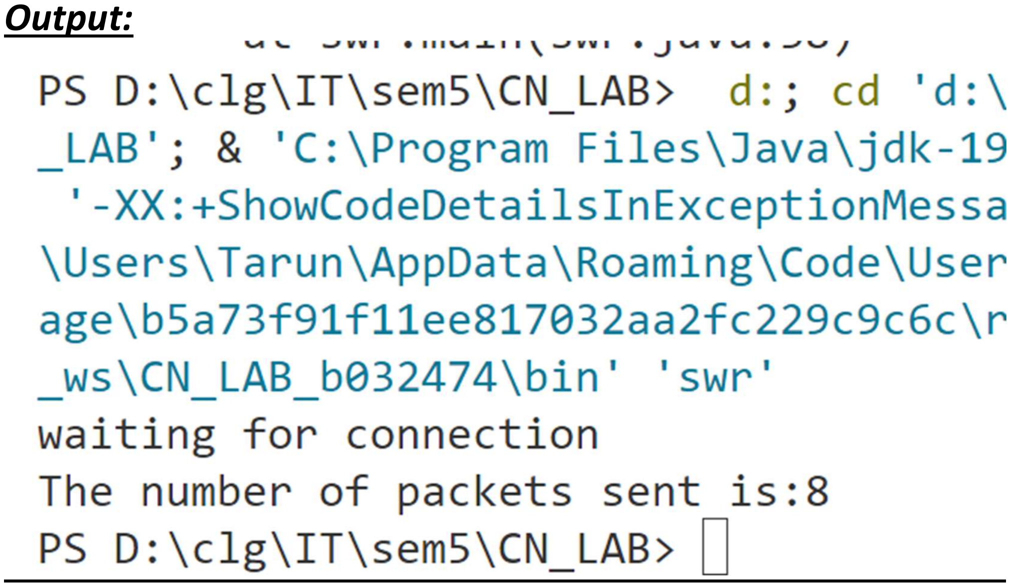
Write a Program to implement Stop and Wait Protocol.

Server Program:

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.SocketException;
public class swr {
   static ServerSocket Serversocket;
   static DataInputStream dis;
   static DataOutputStream dos;
   public static void main(String[] args) throws SocketException {
      try {
         int a[] = { 30, 40, 50, 60, 70, 80, 90, 100 };
         Serversocket = new ServerSocket(1801);
         System.out.println("waiting for connection");
         Socket client = Serversocket.accept();
         dis = new DataInputStream(client.getInputStream());
         dos = new DataOutputStream(client.getOutputStream());
         System.out.println("The number of packets sent is:" + a.length);
         int y = a.length;
         dos.write(y);
         dos.flush();
         for (int i = 0; i < a.length; i++) {
            dos.write(a[i]);
            dos.flush();
         }
         int k = dis.read();
         dos.write(a[k]);
         dos.flush();
      }
```

```
        catch (IOException e){
            System.out.println(e);
        }
        finally{
            try{
                dis.close();
                dos.close();
            }
            catch (IOException e){
                e.printStackTrace();
            }
        }
    }
}
```

Output:

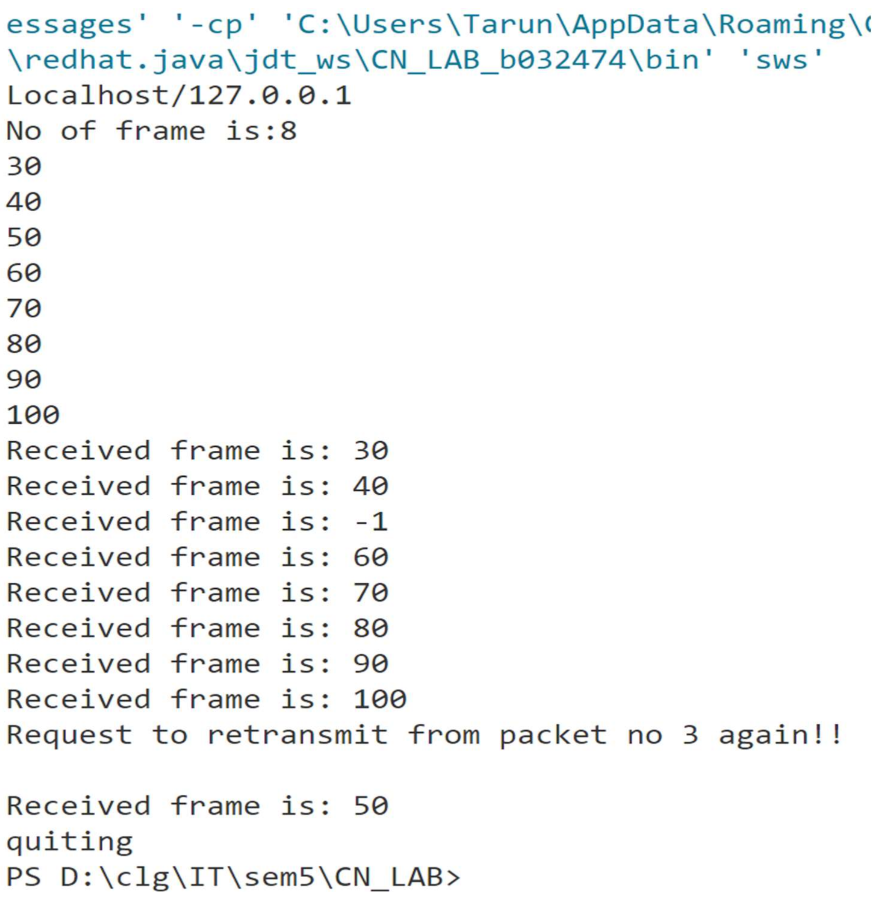


```
PS D:\clg\IT\sem5\CN_LAB>  d:; cd 'd:\
_LAB'; & 'C:\Program Files\Java\jdk-19
 '-XX:+ShowCodeDetailsInExceptionMessa
\Users\Tarun\AppData\Roaming\Code\User
age\b5a73f91f11ee817032aa2fc229c9c6c\r
_ws\CN_LAB_b032474\bin' 'swr'
waiting for connection
The number of packets sent is:8
PS D:\clg\IT\sem5\CN_LAB>
```

Client Program:

```
import java.lang.System;
import java.net.*;
import java.io.*;
import java.util.Random;
public class sws {
   static Socket connection;
   public static void main(String a[]) throws SocketException {
      try {
         int v[] = new int[10];
         int n = 0;
         Random rands = new Random();
         int rand = 0;
         InetAddress addr = InetAddress.getByName("Localhost");
         System.out.println(addr);
         connection = new Socket(addr, 1801);
         DataOutputStream out = new DataOutputStream(
               connection.getOutputStream());
         DataInputStream in = new DataInputStream(
               connection.getInputStream());
         int p = in.read();
         System.out.println("No of frame is:" + p);
         for (int i = 0; i < p; i++) {
            v[i] = in.read();
            System.out.println(v[i]);
         }
         rand = rands.nextInt(p);//FRAME NO. IS RANDOMLY
GENERATED
         v[rand] = -1;
         for (int i = 0; i < p; i++){
               System.out.println("Received frame is: " + v[i]);
         }
         for (int i = 0; i < p; i++)
            if (v[i] == -1) {
                System.out.println("Request to retransmit from packet no "
                    + (i+1) + " again!!");
                n = i;
                out.write(n);
                out.flush();
              }
          System.out.println();
            v[n] = in.read();
            System.out.println("Received frame is: " + v[n]);
          System.out.println("quiting");
        }
        catch (Exception e) {
          System.out.println(e);
        }
      }
    }
```

Output:



```
essages' '-cp' 'C:\Users\Tarun\AppData\Roaming\(
\redhat.java\jdt_ws\CN_LAB_b032474\bin' 'sws'
Localhost/127.0.0.1
No of frame is:8
30
40
50
60
70
80
90
100
Received frame is: 30
Received frame is: 40
Received frame is: -1
Received frame is: 60
Received frame is: 70
Received frame is: 80
Received frame is: 90
Received frame is: 100
Request to retransmit from packet no 3 again!!

Received frame is: 50
quiting
PS D:\clg\IT\sem5\CN_LAB>
```

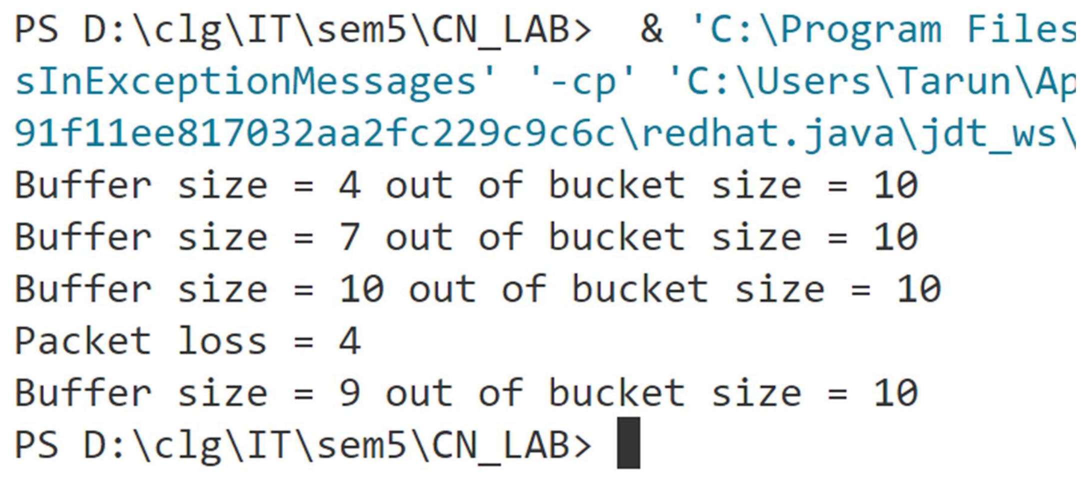
Exercise:9

Write a program for congestion control using leaky bucket algorithm

Program :

```
public class LeakyBucket {
   public static void main(String[] args) {
      int storage = 0;
      int no_of_queries = 4;
      int bucket_size = 10;
      int input_pkt_size = 4;
      int output_pkt_size = 1;
      for (int i = 0; i < no_of_queries; i++) {
         int size_left = bucket_size - storage;
         if (input_pkt_size <= size_left) {
            storage += input_pkt_size;
         } else {
            System.out.println("Packet loss = " + input_pkt_size);
         }
         System.out.println("Buffer size = " + storage + " out of bucket
size = " + bucket_size);
         storage -= output_pkt_size;
      }
   }
}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Files
sInExceptionMessages' '-cp' 'C:\Users\Tarun\Ap
91f11ee817032aa2fc229c9c6c\redhat.java\jdt_ws\
Buffer size = 4 out of bucket size = 10
Buffer size = 7 out of bucket size = 10
Buffer size = 10 out of bucket size = 10
Packet loss = 4
Buffer size = 9 out of bucket size = 10
PS D:\clg\IT\sem5\CN_LAB>
```

Exercise:10

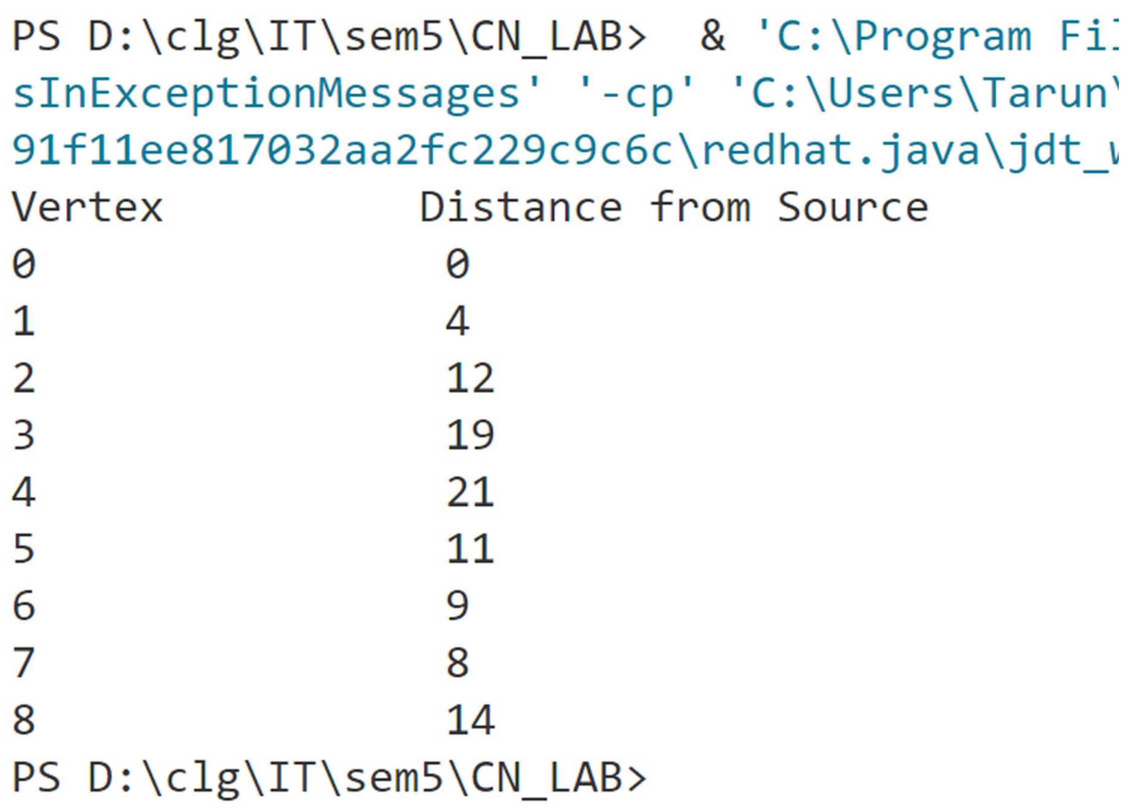
Write a Program to implement Dijkstra's algorithm to compute the Shortest path through a graph.

Program:

```
import java.util.Arrays;

public class dijkk {
   static final int V = 9;

   void dijkstra(int[][] graph, int src) {
      int[] dist = new int[V];
      boolean[] visited = new boolean[V];

      Arrays.fill(dist, Integer.MAX_VALUE);
      dist[src] = 0;

      for (int count = 0; count < V - 1; count++) {
         int u = -1;
         for (int v = 0; v < V; v++) {
            if (!visited[v] && (u == -1 || dist[v] < dist[u])) {
               u = v;
            }
         }

         visited[u] = true;

         for (int v = 0; v < V; v++) {
            if (!visited[v] && graph[u][v] != 0 && dist[u] !=
Integer.MAX_VALUE
                  && dist[u] + graph[u][v] < dist[v]) {
               dist[v] = dist[u] + graph[u][v];
            }
         }
      }
```

```
        printSolution(dist);
    }

    void printSolution(int dist[]) {
        System.out.println("Vertex \t Distance from Source");
        for (int i = 0; i < V; i++) {
            System.out.println(i + " \t " + dist[i]);
        }
    }

    public static void main(String[] args) {
        int graph[][] = {
            {0, 4, 0, 0, 0, 0, 0, 8, 0},
            {4, 0, 8, 0, 0, 0, 11, 0, 0},
            {0, 8, 0, 7, 0, 4, 0, 0, 2},
            {0, 0, 7, 0, 9, 14, 0, 0, 0},
            {0, 0, 0, 9, 0, 10, 0, 0, 0},
            {0, 0, 4, 14, 10, 0, 2, 0, 0},
            {0, 0, 0, 0, 0, 2, 0, 1, 6},
            {8, 11, 0, 0, 0, 0, 1, 0, 7},
            {0, 0, 2, 0, 0, 0, 6, 7, 0}
        };

        DijkstraAlgorithm dijkstra = new DijkstraAlgorithm();
        dijkstra.dijkstra(graph, 0);
    }
}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Fi
sInExceptionMessages' '-cp' 'C:\Users\Tarun
91f11ee817032aa2fc229c9c6c\redhat.java\jdt_
Vertex                Distance from Source
0                      0
1                      4
2                      12
3                      19
4                      21
5                      11
6                      9
7                      8
8                      14
PS D:\clg\IT\sem5\CN_LAB>
```

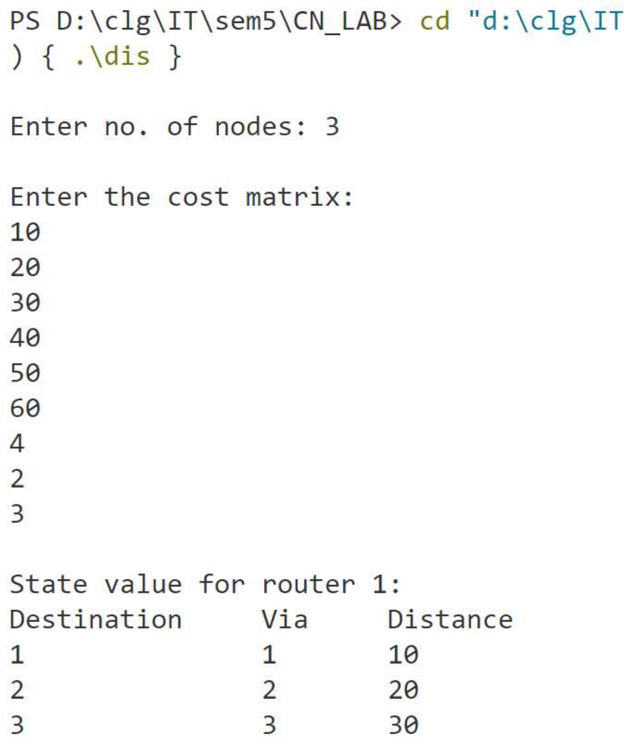
Exercise:11

Write a Program to implement Distance vector routing algorithm by obtaining routing table at each node (Take an example subnet graph with weights indicating delay
between nodes).

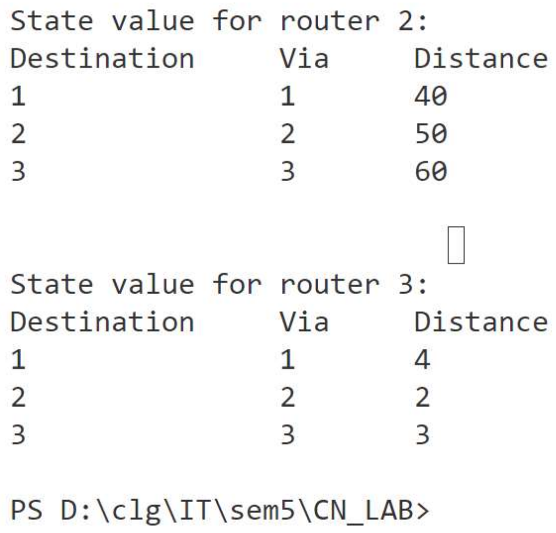
Program:

```
#include<stdio.h>
#include<stdlib.h>
struct node {
   unsigned dist[20];
   unsigned from[20];
};

int main() {
   struct node rt[10]; // Define rt array with the correct type
   int dmat[20][20];
   int n, i, j, k, count = 0;

   printf("\nEnter no. of nodes: ");
   scanf("%d", &n);

   printf("\nEnter the cost matrix:\n");
   for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
         scanf("%d", &dmat[i][j]);
         rt[i].dist[j] = dmat[i][j];
         rt[i].from[j] = j;
      }
   }

   do {
      count = 0;
      for (i = 0; i < n; i++) {
         for (j = 0; j < n; j++) {
            for (k = 0; k < n; k++) {
                if (rt[i].dist[j] > dmat[i][k] + rt[k].dist[j]) {
                    rt[i].dist[j] = rt[i].dist[k] + rt[k].dist[j];
                    rt[i].from[j] = k;
                    count++;
                }
            }
        }
    }
} while (count != 0);

for (i = 0; i < n; i++) {
    printf("\nState value for router %d:\n", i + 1);
    printf("Destination\tVia\tDistance\n");
    for (j = 0; j < n; j++) {
        printf("%d\t\t%d\t%d\n", j + 1, rt[i].from[j] + 1, rt[i].dist[j]);
    }
    printf("\n");
}

return 0;
}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB> cd "d:\clg\IT
) { .\dis }

Enter no. of nodes: 3

Enter the cost matrix:
10
20
30
40
50
60
4
2
3

State value for router 1:
Destination     Via     Distance
1               1       10
2               2       20
3               3       30
```



```
State value for router 2:
Destination     Via     Distance
1               1       40
2               2       50
3               3       60


State value for router 3:
Destination     Via     Distance
1               1       4
2               2       2
3               3       3

PS D:\clg\IT\sem5\CN_LAB>
```

Exercise:12

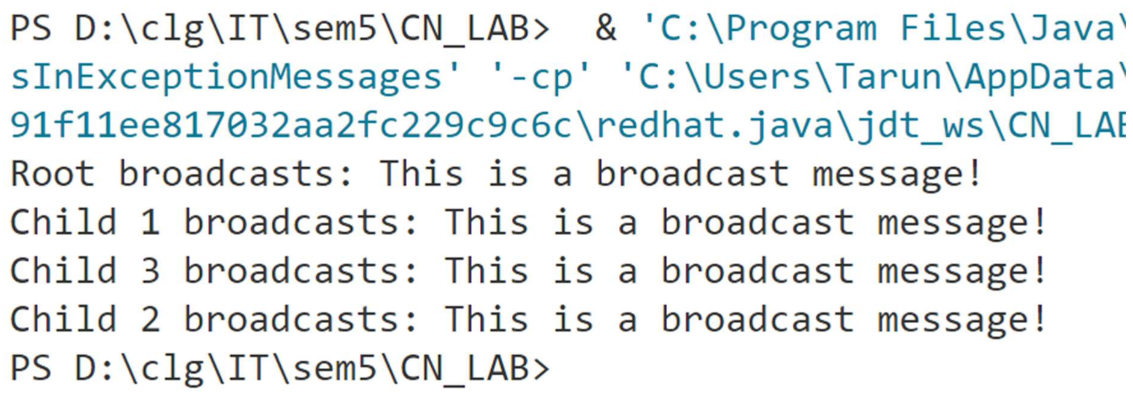
Write a Program to implement Broadcast tree by taking subnet of hosts.

Program:

```
import java.util.ArrayList;
import java.util.List;

class Host {
   private String name;
   private List<Host> children;

   public Host(String name) {
      this.name = name;
      this.children = new ArrayList<>();
   }

   public void addChild(Host child) {
      children.add(child);
   }

   public void broadcastMessage(String message) {
      System.out.println(name + " broadcasts: " + message);
      for (Host child : children) {
         child.broadcastMessage(message);
      }
   }
}

public class BroadcastTree {
   public static void main(String[] args) {
      Host root = new Host("Root");
      Host child1 = new Host("Child 1");
      Host child2 = new Host("Child 2");
```

```
        Host child3 = new Host("Child 3");

        root.addChild(child1);
        root.addChild(child2);
        child1.addChild(child3);

        root.broadcastMessage("This is a broadcast message!");
    }
}
```

Output:



```
PS D:\clg\IT\sem5\CN_LAB>  & 'C:\Program Files\Java\
sInExceptionMessages' '-cp' 'C:\Users\Tarun\AppData\
91f11ee817032aa2fc229c9c6c\redhat.java\jdt_ws\CN_LAE
Root broadcasts: This is a broadcast message!
Child 1 broadcasts: This is a broadcast message!
Child 3 broadcasts: This is a broadcast message!
Child 2 broadcasts: This is a broadcast message!
PS D:\clg\IT\sem5\CN_LAB>
```

Exercise:13

Wireshark

i. Packet Capture Using Wire shark

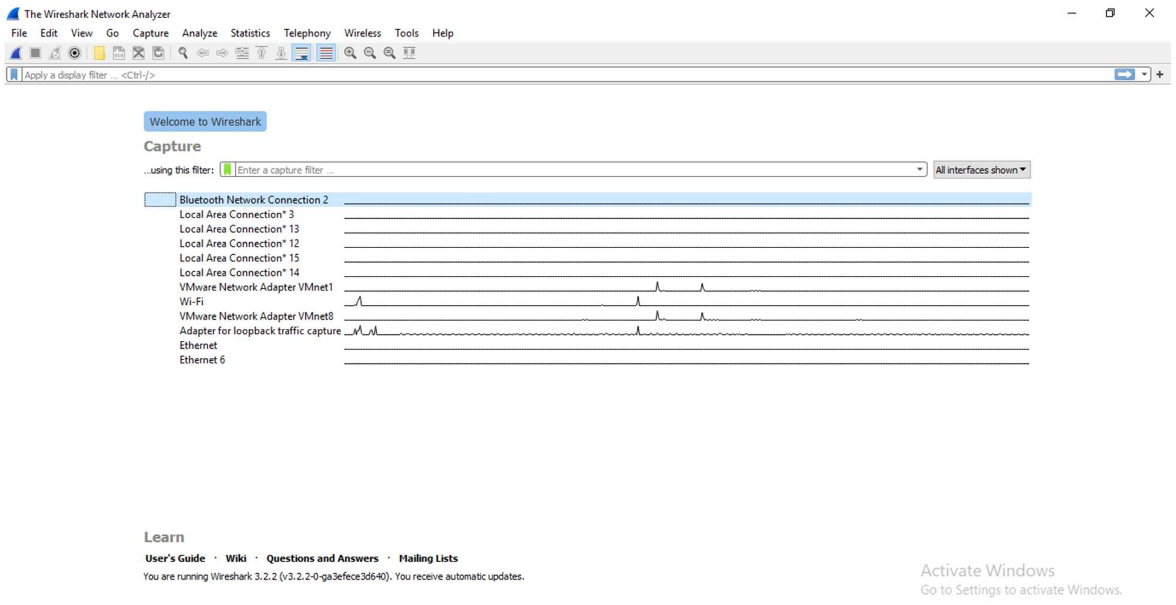
ii. Starting Wire shark

iii. Viewing Captured Traffic

iv. Analysis and Statistics & Filters.

Wireshark captures the data coming or going through the NICs on its device by using an underlying packet capture library. By default, Wireshark captures on-device data only, but it can capture almost all the data on its LAN if run in promiscuous mode. Currently, Wireshark uses NMAP's Packet Capture library(called npcap).

Getting Up and Running: After installation launch Wireshark, approve the administrator or superuser privileges and you will be presented with a window that looks like this:


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
Activate Windows
Go to Settings to activate Windows.

Bluetooth Network Connection 2
Local Area Connection* 3
Local Area Connection* 13
Local Area Connection* 12
Local Area Connection* 15
Local Area Connection* 14
VMware Network Adapter VMnet1
Wi-Fi
VMware Network Adapter VMnet8
Adapter for loopback traffic capture
Ethernet
Ethernet 6

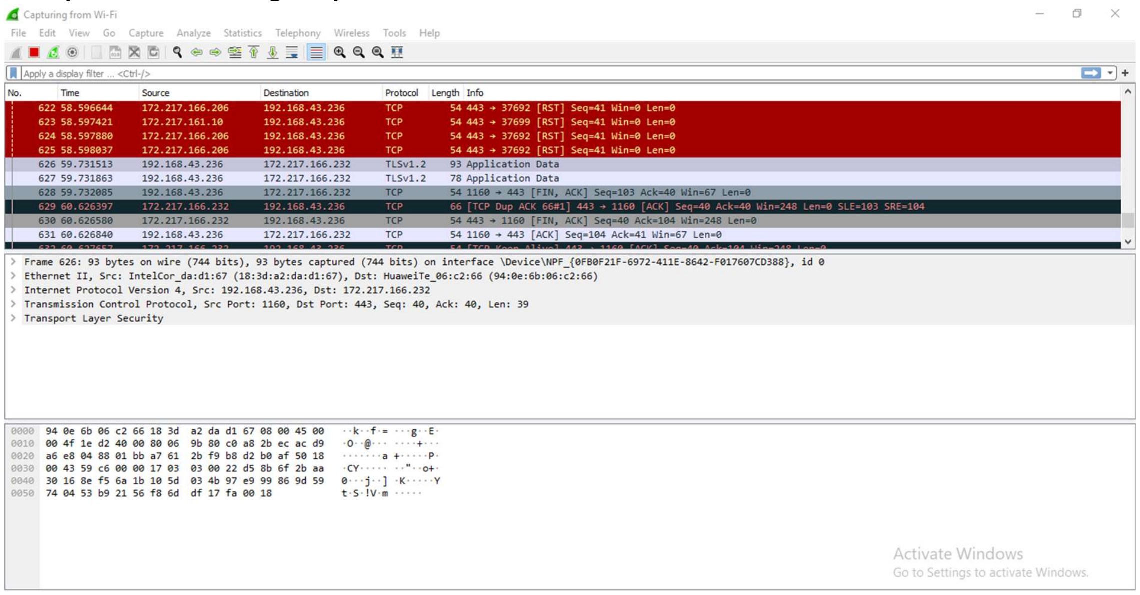
You are running Wireshark 3.2.2 (v3.2.2-0-ga3efece3d640). You receive automatic updates.

Wireshark Launch Screen

This window shows the interfaces on your device. To start sniffing select one interface and click on the bluefin icon on the top left. The data capture screen has three panes. The top pane shows real-time traffic, the middle one shows information about the chosen packet and the bottom pane shows the raw packet data. The top pane shows *source address(IPv4 or IPv6) destination address, source and destination ports, protocol to which the packet belongs to and additional information about the packet.*

<u>Wireshark capture screen</u>

Since there are a lot of packets going in and out every second, looking at all of them or searching for one type of packets will be tedious. This is why packet filters are provided. Packets can be filtered based on many parameters like IP address, port number or protocol at capture level or at display level. As obvious a display level filter will not affect the packets being captured.


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.   Time        Source           Destination      Protocol  Length Info
  622 58.596644   172.217.166.206  192.168.43.236   TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
  623 58.597421   172.217.161.10   192.168.43.236   TCP       54 443 → 37699 [RST] Seq=41 Win=0 Len=0
  624 58.597880   172.217.166.206  192.168.43.236   TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
  625 58.598037   172.217.166.206  192.168.43.236   TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
  626 59.731513   192.168.43.236   172.217.166.232  TLSv1.2   93 Application Data
  627 59.731863   192.168.43.236   172.217.166.232  TLSv1.2   78 Application Data
  628 59.732085   192.168.43.236   172.217.166.232  TCP       54 1160 → 443 [FIN, ACK] Seq=103 Ack=40 Win=67 Len=0
  629 60.626397   172.217.166.232  192.168.43.236   TCP       66 [TCP Dup ACK 66#1] 443 → 1160 [ACK] Seq=40 Ack=40 Win=248 Len=0 SLE=103 SRE=104
  630 60.626580   172.217.166.232  192.168.43.236   TCP       54 443 → 1160 [FIN, ACK] Seq=40 Ack=104 Win=248 Len=0
  631 60.626840   192.168.43.236   172.217.166.232  TCP       54 1160 → 443 [ACK] Seq=104 Ack=41 Win=67 Len=0

> Frame 626: 93 bytes on wire (744 bits), 93 bytes captured (744 bits) on interface \Device\NPF_{0FB0F21F-6972-411E-8642-F017607CD388}, id 0
> Ethernet II, Src: IntelCor_da:d1:67 (18:3d:a2:da:d1:67), Dst: HuaweiTe_06:c2:66 (94:0e:6b:06:c2:66)
> Internet Protocol Version 4, Src: 192.168.43.236, Dst: 172.217.166.232
> Transmission Control Protocol, Src Port: 1160, Dst Port: 443, Seq: 40, Ack: 40, Len: 39
> Transport Layer Security

0000  94 0e 6b 06 c2 66 18 3d  a2 da d1 67 08 00 45 00   ··k··f·=  ···g··E·
0010  00 4f 1e d2 40 00 80 06  9b 80 c0 a8 2b ec ac d9   ·O··@···  ····+···
0020  a6 e8 04 88 01 bb a7 61  2b f9 b8 d2 b0 af 50 18   ·······a  +·····P·
0030  00 43 59 c6 00 00 17 03  03 00 22 d5 8b 6f 2b aa   ·CY·····  ··"··o+·
0040  30 16 8e f5 6a 1b 10 5d  03 4b 97 e9 99 86 9d 59   0···j··]  ·K·····Y
0050  74 04 53 b9 21 56 f8 6d  df 17 fa 00 18            t·S·!V·m  ·····
```

Some of the general capture filters are:

host (capture the traffic through a single target)

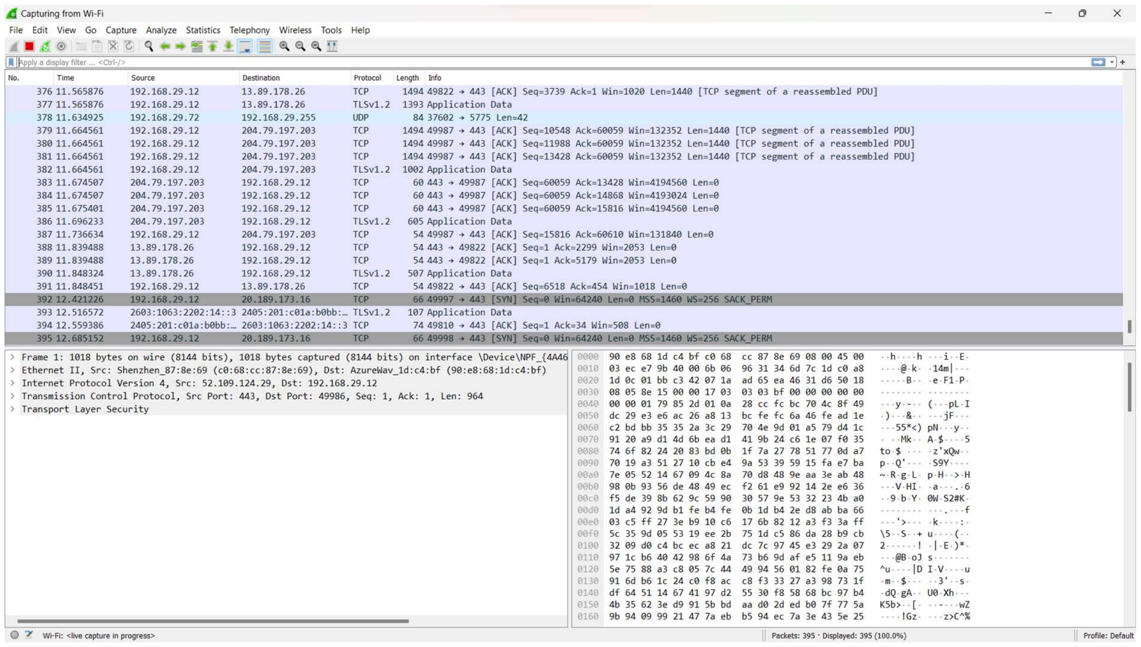
net(capture the traffic through a network or sub-network). “net” can be prefixed with “src” or “dst” to indicate whether the data coming from or going to the target host(s).)

port (capture the traffic through or from a port). “port” can be prefixed with “src” or “dst” to indicate whether the data coming from or going to the target port.

“and”, “not” and “or” logical connectives.(Used to combine multiple filters together).

There are some more basic filters and they can be combined very creatively. Another range of filters, display filters are used to create

abstraction on captured data. These basic examples should provide a basic idea of their syntax:
tcp.port==80/udp.port==X shows the tcp/udp traffic at port X.
http.request.uri matches "parameter=value$" shows packets that are HTTP requests at the application layer level and their URI ends with a parameter with some value.
The logical connective and or and not work here too.
ip.src==192.168.0.0/16 and ip.dst==192.168.0.0/16 will show traffic to and from workstations and servers.
There is also a concept of coloring rules. Each protocol/port/other element is provided a unique color to make it easily visible for quick analysis. More details on coloring rules is here
Plugins are extra pieces of codes that can be embedded into the native Wireshark. Plugins help in analysis by:


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time       Source                 Destination            Protocol Length Info
 376 11.565876  192.168.29.12          13.89.178.26           TCP      1494 49822 → 443 [ACK] Seq=3739 Ack=1 Win=1020 Len=1440 [TCP segment of a reassembled PDU]
 377 11.565876  192.168.29.12          13.89.178.26           TLSv1.2  1393 Application Data
 378 11.634925  192.168.29.72          192.168.29.255         UDP        84 37602 → 5775 Len=42
 379 11.664561  192.168.29.12          204.79.197.203         TCP      1494 49987 → 443 [ACK] Seq=10548 Ack=60059 Win=132352 Len=1440 [TCP segment of a reassembled PDU]
 380 11.664561  192.168.29.12          204.79.197.203         TCP      1494 49987 → 443 [ACK] Seq=11988 Ack=60059 Win=132352 Len=1440 [TCP segment of a reassembled PDU]
 381 11.664561  192.168.29.12          204.79.197.203         TCP      1494 49987 → 443 [ACK] Seq=13428 Ack=60059 Win=132352 Len=1440 [TCP segment of a reassembled PDU]
 382 11.664561  192.168.29.12          204.79.197.203         TLSv1.2  1002 Application Data
 383 11.674507  204.79.197.203         192.168.29.12          TCP        60 443 → 49987 [ACK] Seq=60059 Ack=13428 Win=4194560 Len=0
 384 11.674507  204.79.197.203         192.168.29.12          TCP        60 443 → 49987 [ACK] Seq=60059 Ack=14868 Win=4193024 Len=0
 385 11.675401  204.79.197.203         192.168.29.12          TCP        60 443 → 49987 [ACK] Seq=60059 Ack=15816 Win=4194560 Len=0
 386 11.696233  204.79.197.203         192.168.29.12          TLSv1.2   605 Application Data
 387 11.736634  192.168.29.12          204.79.197.203         TCP        54 49987 → 443 [ACK] Seq=15816 Ack=60610 Win=131840 Len=0
 388 11.839488  13.89.178.26           192.168.29.12          TCP        54 443 → 49822 [ACK] Seq=1 Ack=2299 Win=2053 Len=0
 389 11.839488  13.89.178.26           192.168.29.12          TCP        54 443 → 49822 [ACK] Seq=1 Ack=5179 Win=2053 Len=0
 390 11.848324  13.89.178.26           192.168.29.12          TLSv1.2   507 Application Data
 391 11.848451  192.168.29.12          13.89.178.26           TCP        54 49822 → 443 [ACK] Seq=6518 Ack=454 Win=1018 Len=0
 392 12.421226  192.168.29.12          20.189.173.16          TCP        66 49997 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
 393 12.516572  2603:1063:2202:14::3   2405:201:c01a:b0bb:…   TLSv1.2   107 Application Data
 394 12.559386  2405:201:c01a:b0bb:…   2603:1063:2202:14::3   TCP        74 49810 → 443 [ACK] Seq=1 Ack=34 Win=508 Len=0
 395 12.685152  192.168.29.12          20.189.173.16          TCP        66 49998 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
```

```
> Frame 1: 1018 bytes on wire (8144 bits), 1018 bytes captured (8144 bits) on interface \Device\NPF_{4A46
> Ethernet II, Src: Shenzhen_87:8e:69 (c0:68:cc:87:8e:69), Dst: AzureWav_1d:c4:bf (90:e8:68:1d:c4:bf)
> Internet Protocol Version 4, Src: 52.109.124.29, Dst: 192.168.29.12
> Transmission Control Protocol, Src Port: 443, Dst Port: 49986, Seq: 1, Ack: 1, Len: 964
> Transport Layer Security
```

```
0000  90 e8 68 1d c4 bf c0 68  cc 87 8e 69 08 00 45 00   ··h····h ···i··E·
0010  03 ec e7 9b 40 00 6b 06  96 31 34 6d 7c 1d c0 a8   ····@·k· ·14m|···
0020  1d 0c 01 bb c3 42 07 1a  ad 65 ea 46 31 d6 50 18   ·····B·· ·e·F1·P·
0030  08 05 8e 15 00 00 17 03  03 03 bf 00 00 00 00 00   ········ ········
0040  00 00 01 79 85 2d 01 0a  28 cc fc bc 70 4c 8f 49   ···y·-·· (···pL·I
0050  dc 29 e3 e6 ac 26 a8 13  bc fe fc 6a 46 fe ad 1e   ·)···&·· ···jF···
0060  c2 bd bb 35 35 2a 3c 29  70 4e 9d 01 a5 79 d4 1c   ···55*<) pN···y··
0070  91 20 a9 d1 4d 6b ea d1  41 9b 24 c6 1e 07 f0 35   · ··Mk·· A·$····5
0080  74 6f 82 24 20 83 bd 0b  1f 7a 27 78 51 77 0d a7   to·$ ··· ·z'xQw··
0090  70 19 a3 51 27 10 cb e4  9a 53 39 59 15 fa e7 ba   p··Q'··· ·S9Y····
00a0  7e 05 52 14 67 09 4c 8a  70 d8 48 9e aa 3e ab 48   ~·R·g·L· p·H··>·H
00b0  98 0b 93 56 de 48 49 ec  f2 61 e9 92 14 2e e6 36   ···V·HI· ·a···.·6
00c0  f5 de 39 8b 62 9c 59 90  30 57 9e 53 32 23 4b a0   ··9·b·Y· 0W·S2#K·
00d0  1d a4 92 9d b1 fe b4 fe  0b 1d b4 2e d8 ab ba 66   ········ ···.···f
00e0  03 c5 ff 27 3e b9 10 c6  17 6b 82 12 a3 f3 3a ff   ···'>··· ·k····:·
00f0  5c 35 9d 05 53 19 ee 2b  75 1d c5 86 da 28 b9 cb   \5··S··+ u····(··
0100  32 09 d0 c4 bc ec a8 21  dc 7c 97 45 e3 29 2a 07   2······! ·|·E·)*·
0110  97 1c b6 40 42 98 6f 4a  73 b6 9d af e5 11 9a eb   ···@B·oJ s·······
0120  5e 75 88 a3 c8 05 7c 44  49 94 56 01 82 fe 0a 75   ^u····|D I·V····u
0130  91 6d b6 1c 24 c0 f8 ac  c8 f3 33 27 a3 98 73 1f   ·m··$··· ··3'··s·
0140  df 64 51 14 67 41 97 d2  55 30 f8 58 68 bc 97 b4   ·dQ·gA·· U0·Xh···
0150  4b 35 62 3e d9 91 5b bd  aa d0 2d ed b0 7f 77 5a   K5b>··[· ··-···wZ
0160  9b 94 09 99 21 47 7a eb  b5 94 ec 7a 3e 43 5e 25   ····!Gz· ···z>C^%
```

Wi-Fi: <live capture in progress>
Packets: 395 · Displayed: 395 (100.0%)
Profile: Default

Showing parameter specific statistics and insights.
Handling capture files and issues related to their formats.
Collaborating with other tools and frameworks to set up an all-in-one network monitoring solution.

With just the basic capability to see all the traffic going through your device or in your LAN and the tools and plugins to help you in analysis, you can do a great deal of things with your device. Like:

- *Troubleshooting Internet connectivity problems with your device or WiFi.*
- *Monitoring your device for unwanted traffic that may be an indication of a malware infection.*
- *Testing the working of your application that involve networking.*
- *Using it to just understand how computer networks work.*

Exercise:14

How to run Nmap scan

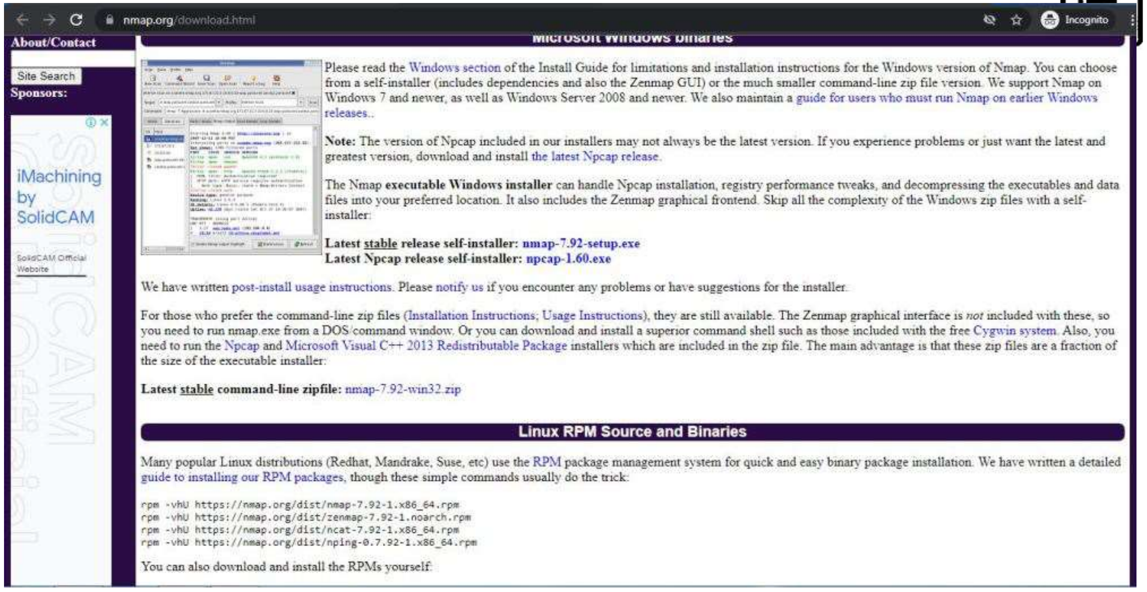
Nmap is computer software that is used to scan networks. It was developed by Gordon Lyon. It is written in C, C++, Python, and Lua. Its initial release was in 1997, and its stable release was in 2021. Its latest version is 7.92. It is free software used for security purposes of networks. It can be run on different operating systems like Windows, Mac, Linux, etc. It is used to protect computers by discovering hosts and services on a computer network by sending data packets and analyzing the responses. Some of the basic features of Nmap are:

- *It discovers hosts on a network.*
- *It also scans open ports on the target hosts.*
- *It also finds the application name and the version number by interacting with network services on remote devices.*
- *It also finds the OS and hardware characteristics of the network devices.*
- *It also does scriptable interaction with the target with the help of NSE(Nmap Scripting Engine) and Lua programming languages.*
- *It is open-source software and is available for most operating systems.*

Installing Nmap on Windows

Follow the below steps to install Nmap on Windows:

***Step 1:** Visit the official website using the URL https://nmap.org/download.html on any web browser the click on **nmap-7.92-setup.exe**. Downloading of this executable file will start soon. It is a 21.8 MB file so it will take some minutes.*



Please read the Windows section of the Install Guide for limitations and installation instructions for the Windows version of Nmap. You can choose from a self-installer (includes dependencies and also the Zenmap GUI) or the much smaller command-line zip file version. We support Nmap on Windows 7 and newer, as well as Windows Server 2008 and newer. We also maintain a guide for users who must run Nmap on earlier Windows releases..

Note: The version of Npcap included in our installers may not always be the latest version. If you experience problems or just want the latest and greatest version, download and install the latest Npcap release.

The Nmap **executable Windows installer** can handle Npcap installation, registry performance tweaks, and decompressing the executables and data files into your preferred location. It also includes the Zenmap graphical frontend. Skip all the complexity of the Windows zip files with a self-installer:

Latest stable release self-installer: nmap-7.92-setup.exe
Latest Npcap release self-installer: npcap-1.60.exe

We have written post-install usage instructions. Please notify us if you encounter any problems or have suggestions for the installer.

For those who prefer the command-line zip files (Installation Instructions; Usage Instructions), they are still available. The Zenmap graphical interface is *not* included with these, so you need to run nmap.exe from a DOS/command window. Or you can download and install a superior command shell such as those included with the free Cygwin system. Also, you need to run the Npcap and Microsoft Visual C++ 2013 Redistributable Package installers which are included in the zip file. The main advantage is that these zip files are a fraction of the size of the executable installer:

Latest stable command-line zipfile: nmap-7.92-win32.zip

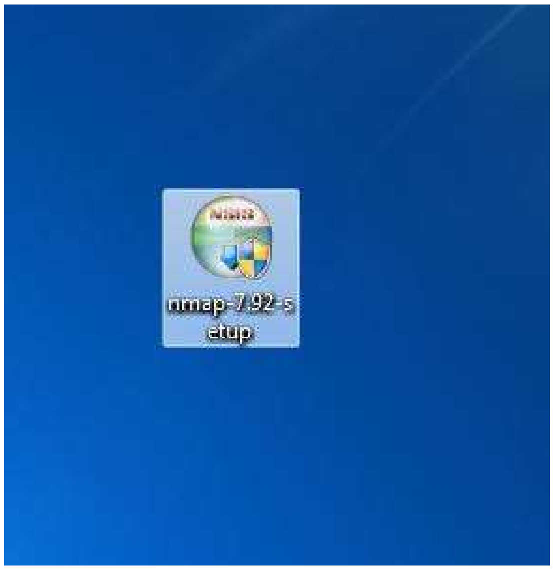
Linux RPM Source and Binaries

Many popular Linux distributions (Redhat, Mandrake, Suse, etc) use the RPM package management system for quick and easy binary package installation. We have written a detailed guide to installing our RPM packages, though these simple commands usually do the trick:

```
rpm -vhU https://nmap.org/dist/nmap-7.92-1.x86_64.rpm
rpm -vhU https://nmap.org/dist/zenmap-7.92-1.noarch.rpm
rpm -vhU https://nmap.org/dist/ncat-7.92-1.x86_64.rpm
rpm -vhU https://nmap.org/dist/nping-0.7.92-1.x86_64.rpm
```

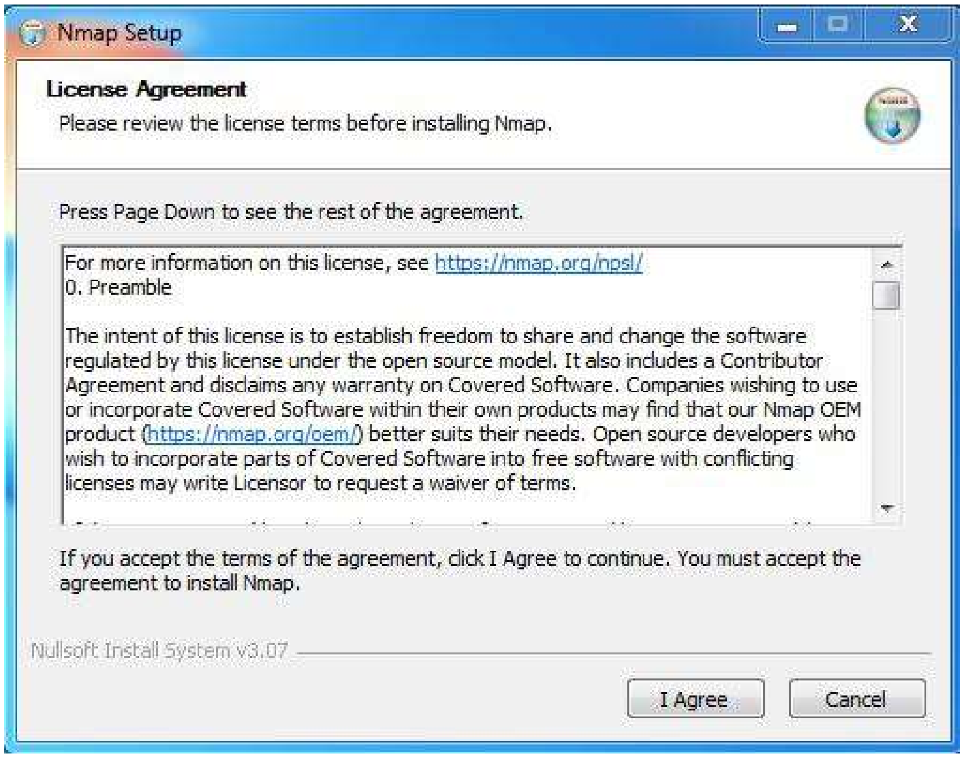
You can also download and install the RPMs yourself:

Step 2: *Now check for the executable file in downloads in your system and run it.*



Step 3: *It will prompt confirmation to make changes to your system. Click on* ***Yes****.*

Step 4: *The next screen will be of License Agreement, click on* ***I Agree****.*



License Agreement

Please review the license terms before installing Nmap.

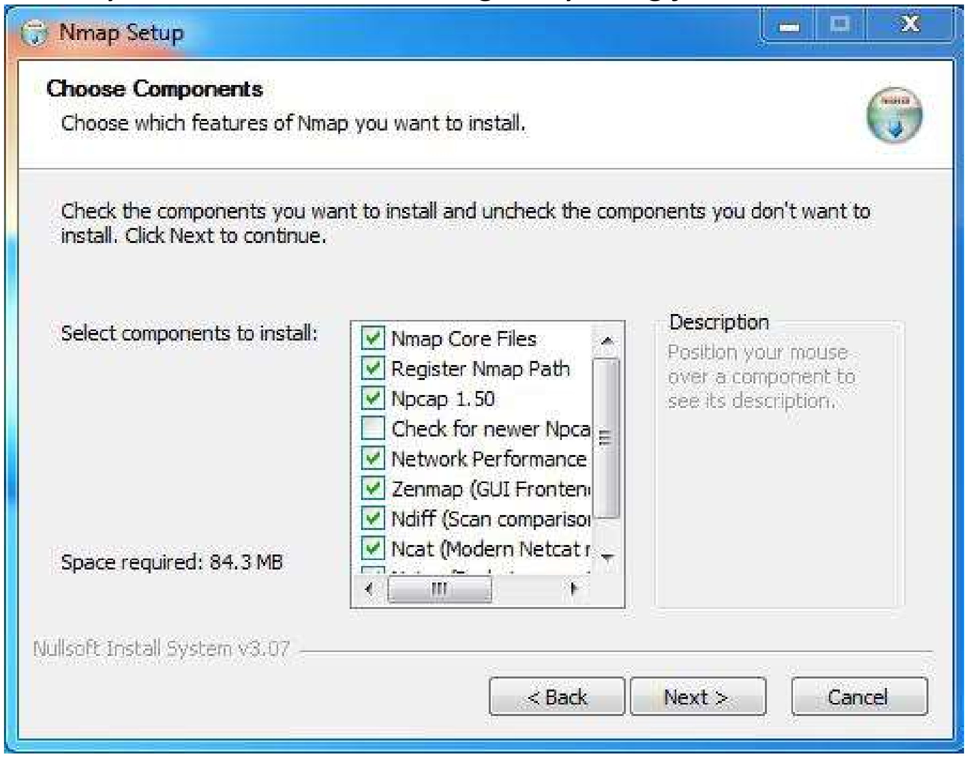
Press Page Down to see the rest of the agreement.

For more information on this license, see https://nmap.org/npsl/
0. Preamble

The intent of this license is to establish freedom to share and change the software regulated by this license under the open source model. It also includes a Contributor Agreement and disclaims any warranty on Covered Software. Companies wishing to use or incorporate Covered Software within their own products may find that our Nmap OEM product (https://nmap.org/oem/) better suits their needs. Open source developers who wish to incorporate parts of Covered Software into free software with conflicting licenses may write Licensor to request a waiver of terms.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Nmap.

Step 5: *Next screen is of choosing components, all components are already marked so don't change anything just click on the* ***Next*** *button.*



Choose Components

Choose which features of Nmap you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

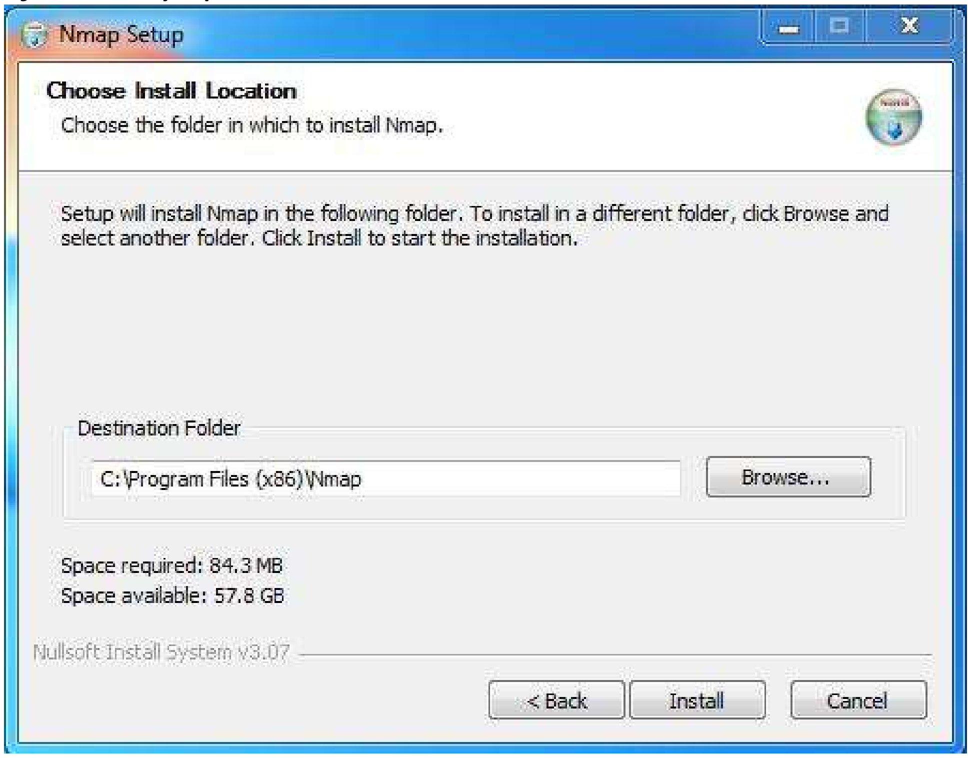
- [x] Nmap Core Files
- [x] Register Nmap Path
- [x] Npcap 1.50
- [ ] Check for newer Npca
- [x] Network Performance
- [x] Zenmap (GUI Fronten
- [x] Ndiff (Scan compariso
- [x] Ncat (Modern Netcat r

Space required: 84.3 MB

Description

Position your mouse over a component to see its description.

Step 6: *In this step, we choose the installation location of Nmap. By default, it uses the C drive but you can change it into another drive that* *will have sufficient memory space for installation. It requires 84.3 MB of memory space.*



Nmap Setup

Choose Install Location

Choose the folder in which to install Nmap.

Setup will install Nmap in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder

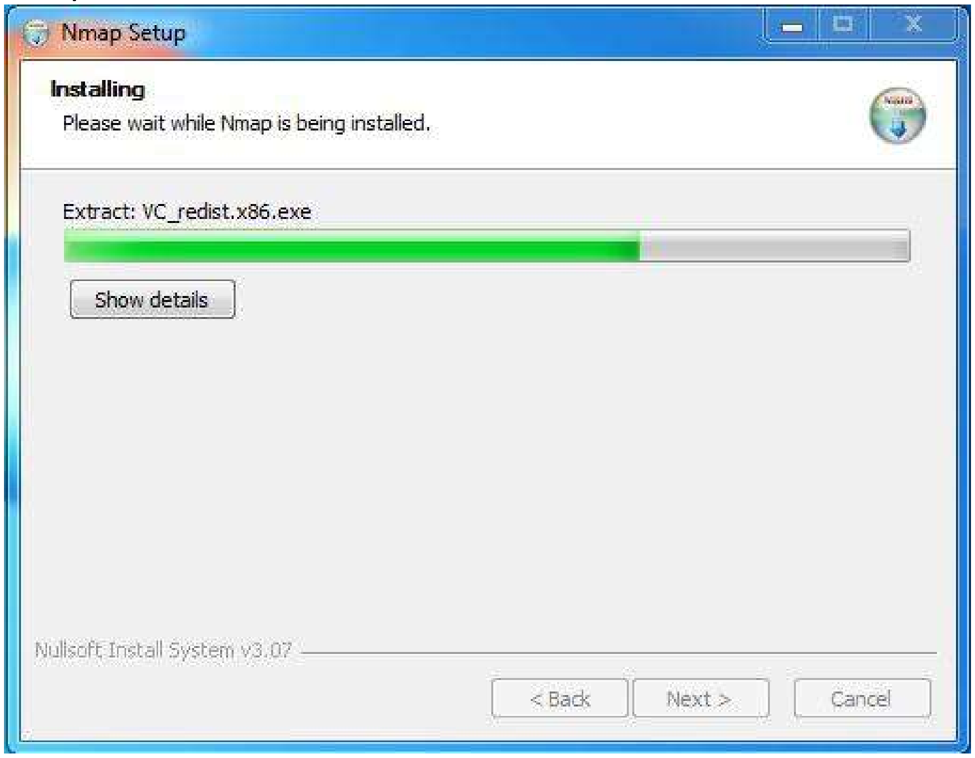
C:\Program Files (x86)\Nmap

Space required: 84.3 MB

Space available: 57.8 GB

Nullsoft Install System v3.07

Step 7: *After this installation process it will take a few minutes to complete the installation.*



Nmap Setup

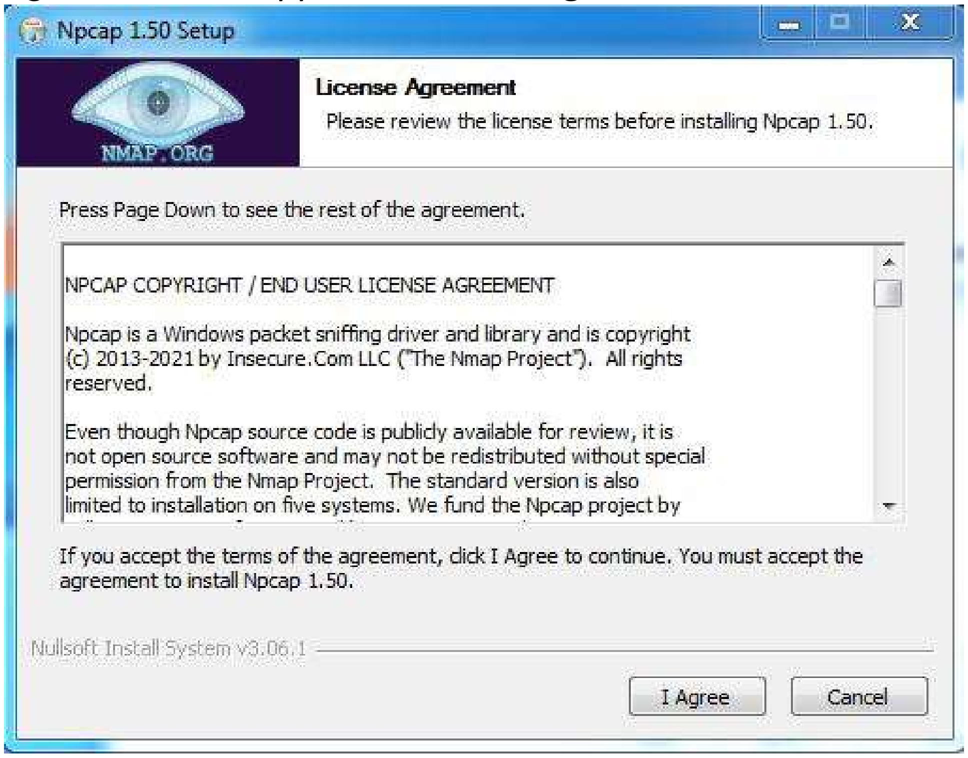
Installing

Please wait while Nmap is being installed.

Extract: VC_redist.x86.exe

Nullsoft Install System v3.07

Step 8: *Npcap installation will also occur with it, the screen of License Agreement will appear, click on* ***I Agree****.*



License Agreement

Please review the license terms before installing Npcap 1.50.

Press Page Down to see the rest of the agreement.

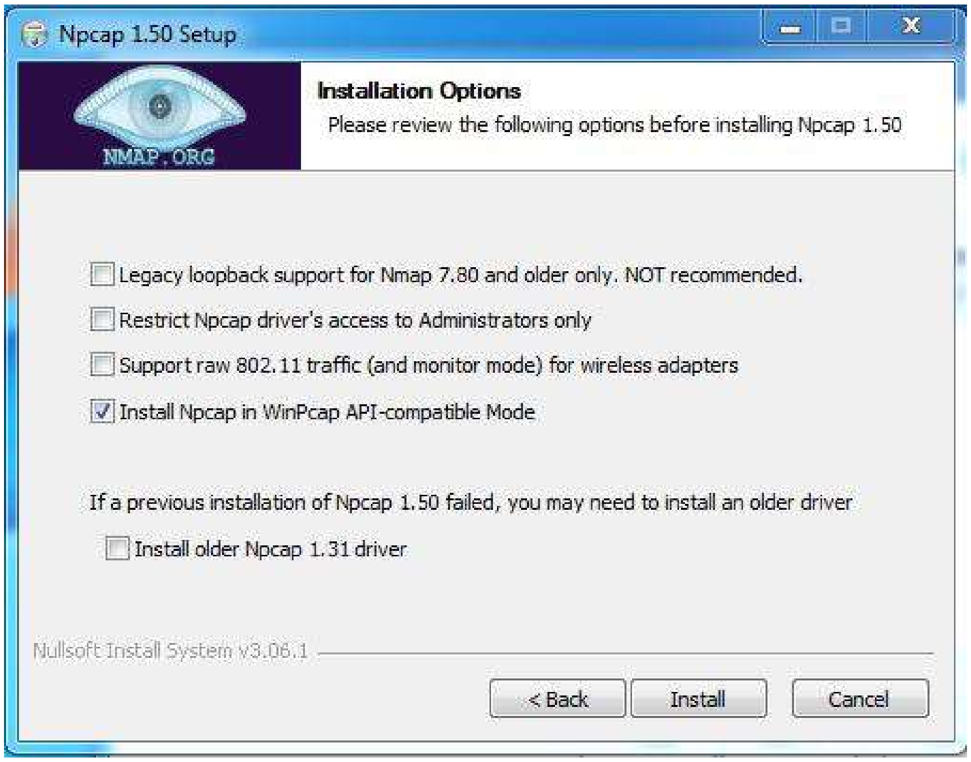
NPCAP COPYRIGHT / END USER LICENSE AGREEMENT

Npcap is a Windows packet sniffing driver and library and is copyright (c) 2013-2021 by Insecure.Com LLC ("The Nmap Project"). All rights reserved.

Even though Npcap source code is publicly available for review, it is not open source software and may not be redistributed without special permission from the Nmap Project. The standard version is also limited to installation on five systems. We fund the Npcap project by

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Npcap 1.50.

Step 9: *Next screen is of installation options don't change anything and click on the* ***Install*** *button.*



Installation Options

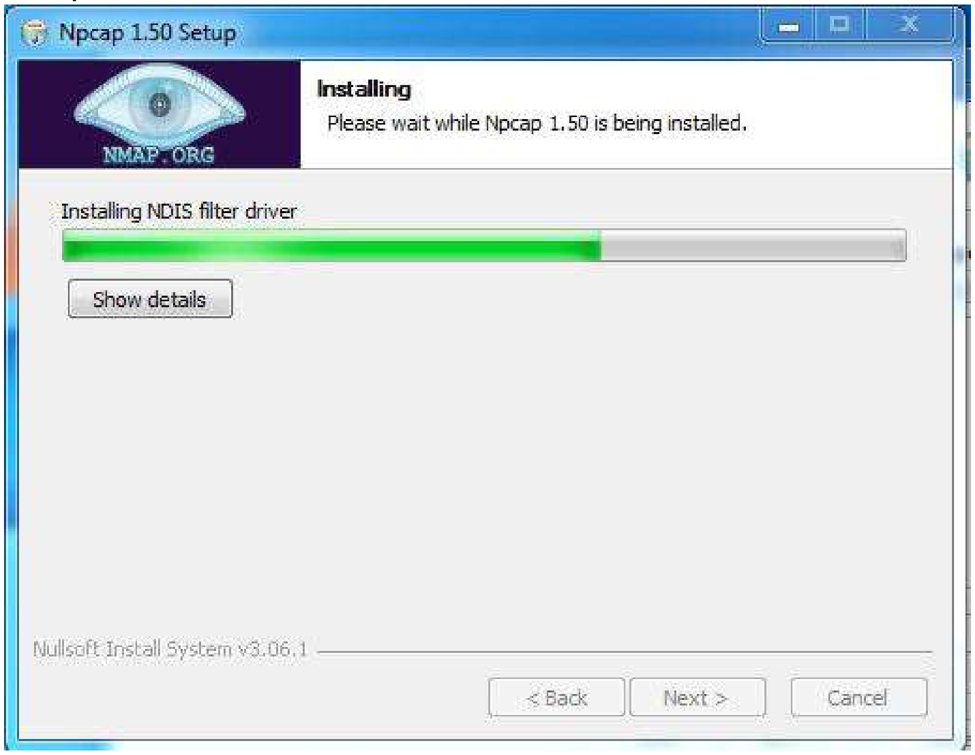
Please review the following options before installing Npcap 1.50

- [ ] Legacy loopback support for Nmap 7.80 and older only. NOT recommended.
- [ ] Restrict Npcap driver's access to Administrators only
- [ ] Support raw 802.11 traffic (and monitor mode) for wireless adapters
- [✓] Install Npcap in WinPcap API-compatible Mode

If a previous installation of Npcap 1.50 failed, you may need to install an older driver

- [ ] Install older Npcap 1.31 driver

Step 10: *After this installation process it will take a few minutes to complete the installation.*

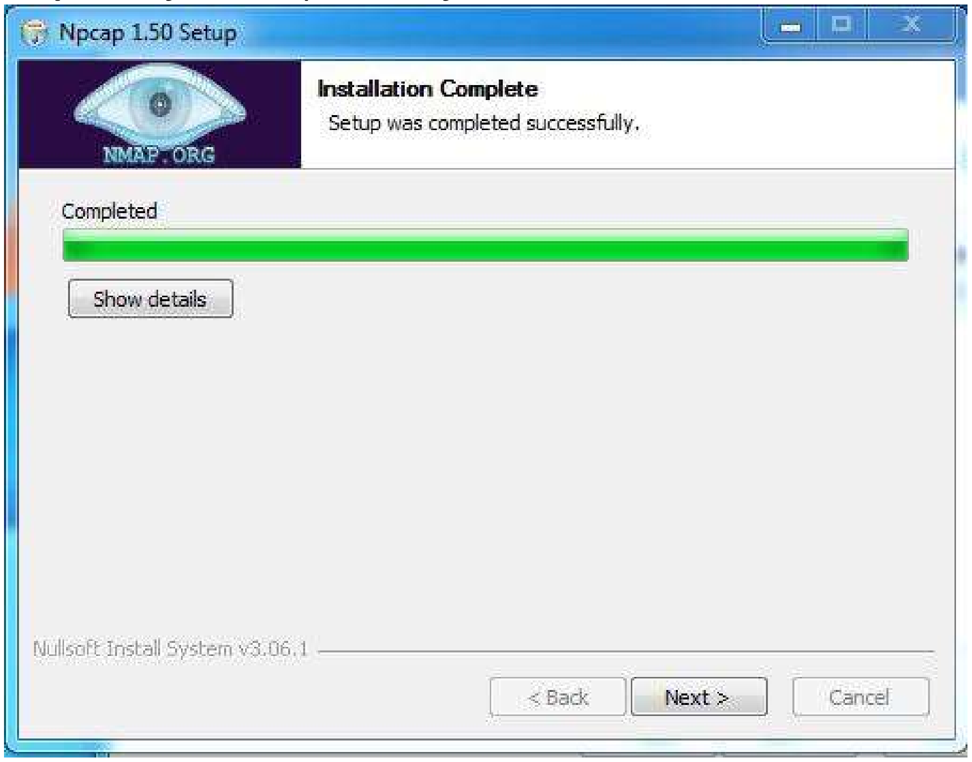

Npcap 1.50 Setup

Installing

Please wait while Npcap 1.50 is being installed.

Installing NDIS filter driver

Step 11: *After completion of installation click on the* ***Next*** *button.*

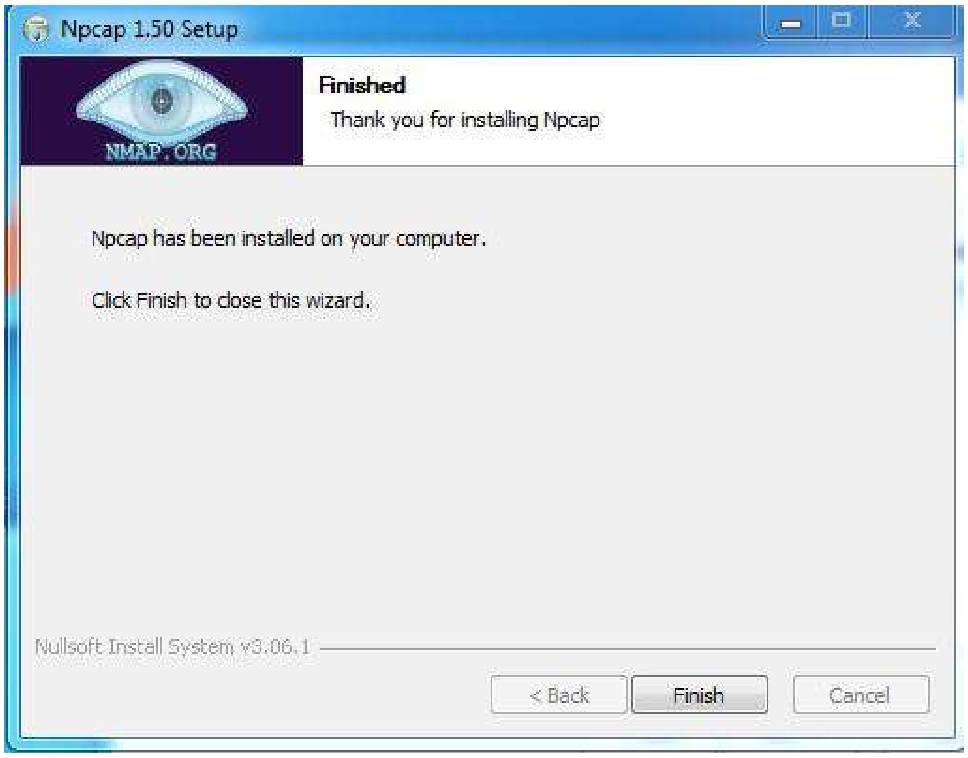

Npcap 1.50 Setup

Installation Complete

Setup was completed successfully.

Completed

Step 12: *Click on the* ***Finish*** *button to finish the installation of Npcap.*



Finished

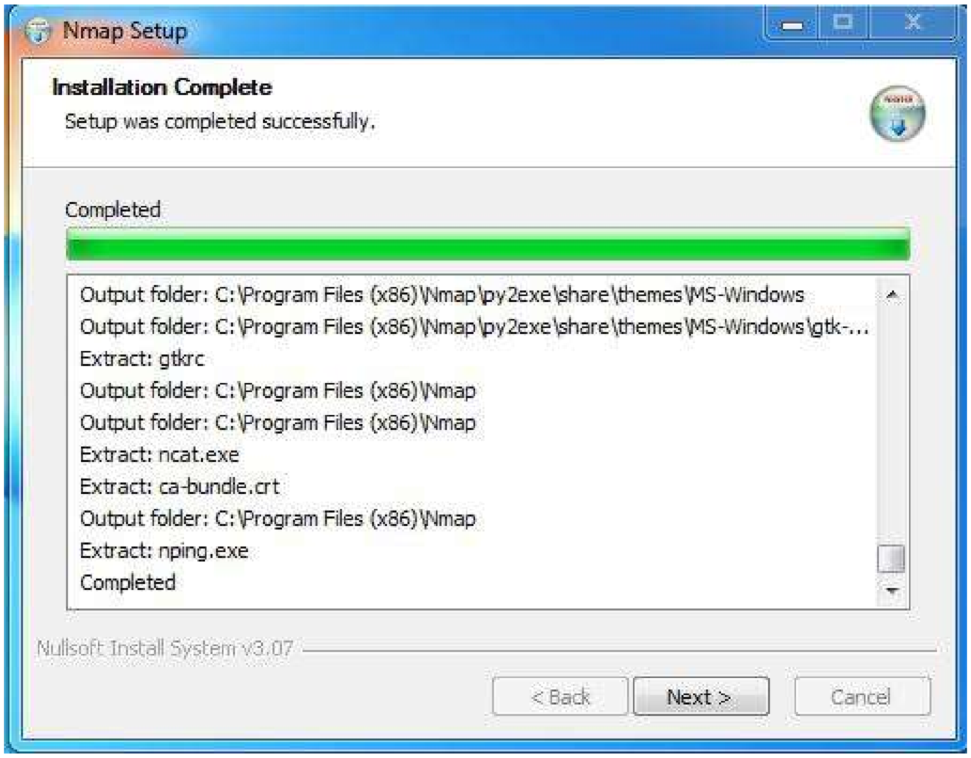
Thank you for installing Npcap

Npcap has been installed on your computer.

Click Finish to close this wizard.

Nullsoft Install System v3.06.1

Step 13: *After completion of the installation of Nmap click on **Next** button.*



Installation Complete

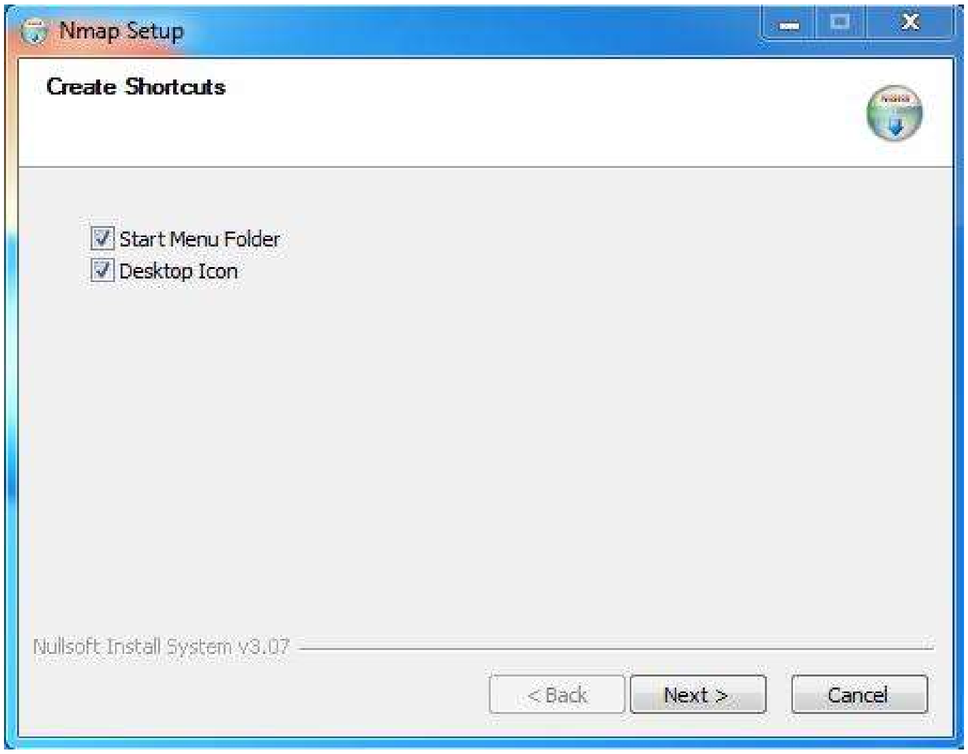
Setup was completed successfully.

Completed

```
Output folder: C:\Program Files (x86)\Nmap\py2exe\share\themes\MS-Windows
Output folder: C:\Program Files (x86)\Nmap\py2exe\share\themes\MS-Windows\gtk-...
Extract: gtkrc
Output folder: C:\Program Files (x86)\Nmap
Output folder: C:\Program Files (x86)\Nmap
Extract: ncat.exe
Extract: ca-bundle.crt
Output folder: C:\Program Files (x86)\Nmap
Extract: nping.exe
Completed
```

Nullsoft Install System v3.07

Step 14: *Screen for creating shortcut will appear, click on **Next** button.*



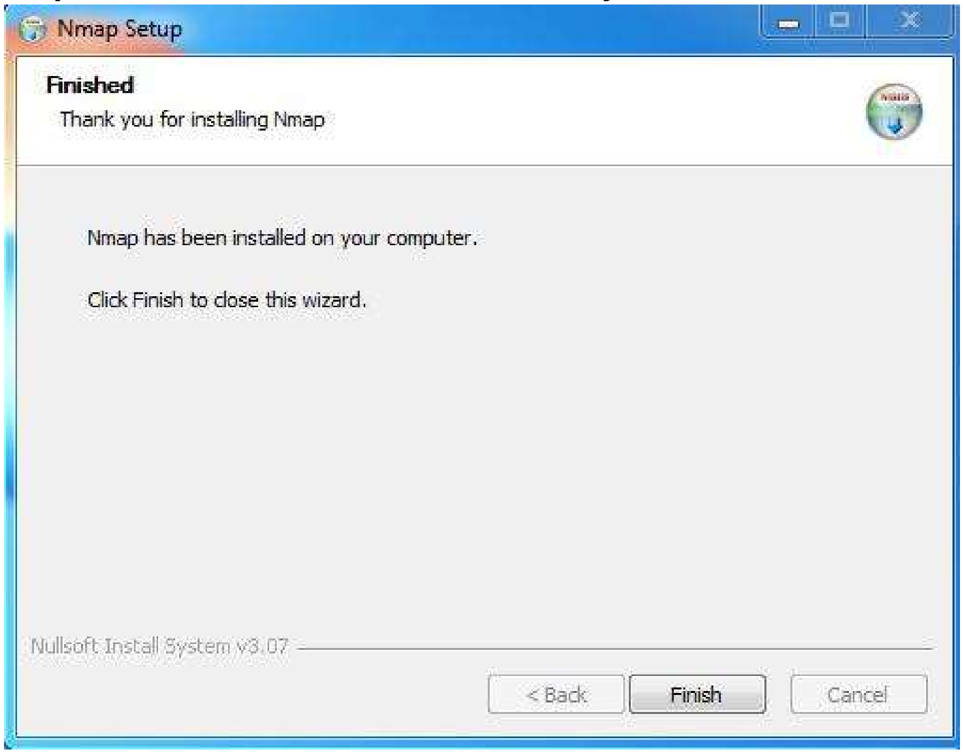
Nmap Setup

Create Shortcuts

☑ Start Menu Folder
☑ Desktop Icon

Nullsoft Install System v3.07

Step 15: *Click on the **Finish** button to finish the installation of Nmap.*



Nmap Setup

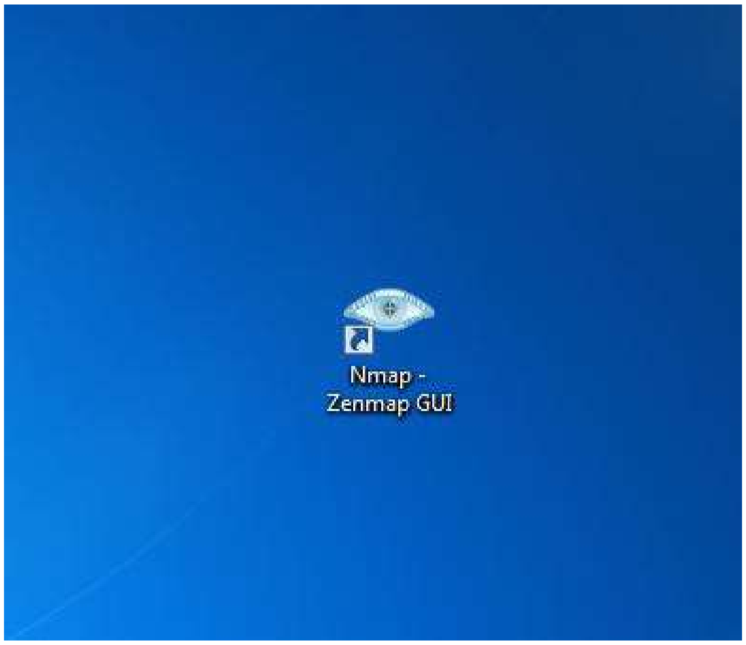
Finished
Thank you for installing Nmap

Nmap has been installed on your computer.

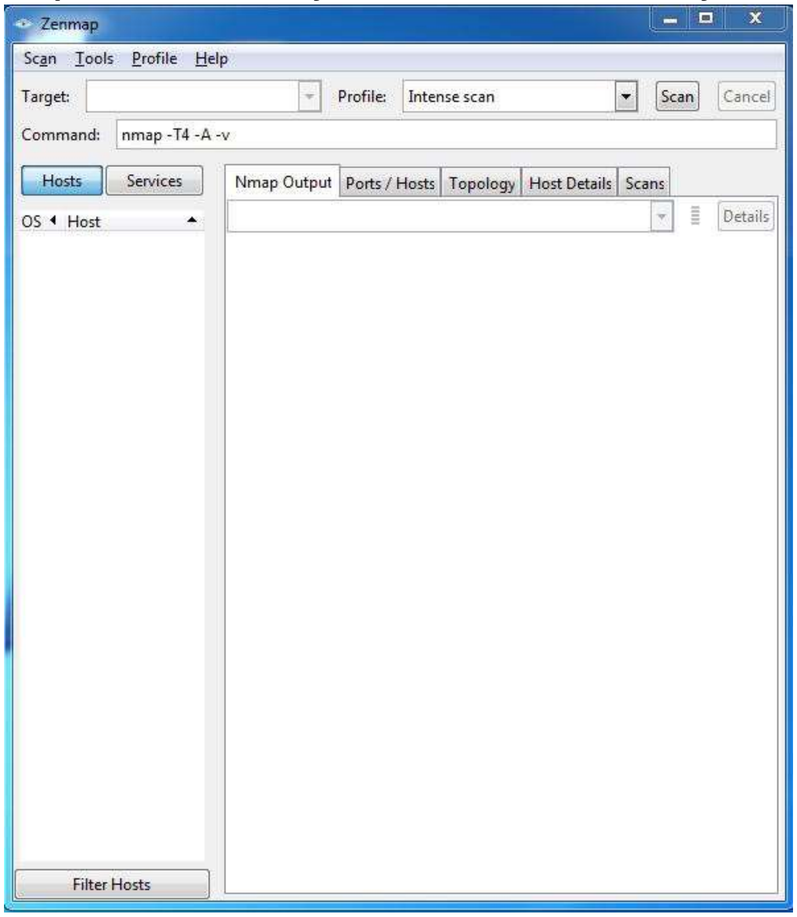
Click Finish to close this wizard.

Nullsoft Install System v3.07

Step 16: *Nmap is successfully installed on the system and an icon is created on the desktop.*



Step 17: *Run the software and see the interface.*



Target:
Profile: Intense scan
Command: nmap -T4 -A -v

So this is how you have successfully installed Nmap on your windows system.

Exercise:15

Operating System Detection using Nmap

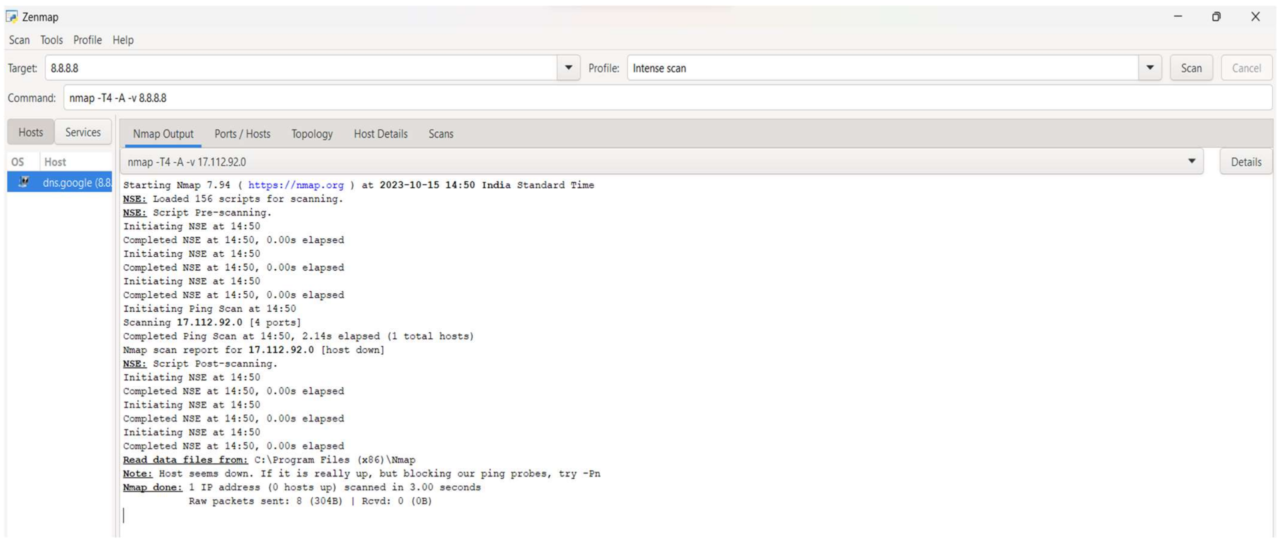
OS Detection

One of Nmap's best-known features is remote OS detection using TCP/IP stack fingerprinting. Nmap sends a series of TCP and UDP packets to the remote host and examines practically every bit in the responses. After performing dozens of tests such as TCP ISN sampling, TCP options support and ordering, IP ID sampling, and the initial window size check, Nmap compares the results to its nmap-os-db database of more than 2,600 known OS fingerprints and prints out the OS details if there is a match. Each fingerprint includes a freeform textual description of the OS, and a classification which provides the vendor name (e.g. Sun), underlying OS (e.g. Solaris), OS generation (e.g. 10), and device type (general purpose, router, switch, game console, etc). Most fingerprints also have a Common Platform Enumeration (CPE) representation, like cpe:/o:linux:linux_kernel:2.6.

If Nmap is unable to guess the OS of a machine, and conditions are good (e.g. at least one open port and one closed port were found), Nmap will provide a URL you can use to submit the fingerprint if you know (for sure) the OS running on the machine. By doing this you contribute to the pool of operating systems known to Nmap and thus it will be more accurate for everyone.

OS detection enables some other tests which make use of information that is gathered during the process anyway. One of these is TCP Sequence Predictability Classification. This measures approximately how hard it is to establish a forged TCP connection against the remote host. It is useful for exploiting source-IP based trust relationships (rlogin, firewall filters, etc) or for hiding the source of an attack. This sort of spoofing is rarely performed any more, but many machines are still vulnerable to it. The actual difficulty number is based on statistical sampling and may fluctuate. It is generally better to use the English classification such as "worthy challenge" or "trivial joke". This is only *reported in normal output in verbose (-v) mode. When verbose mode is enabled along with -O, IP ID sequence generation is also reported. Most machines are in the "incremental" class, which means that they increment the ID field in the IP header for each packet they send. This makes them vulnerable to several advanced information gathering and spoofing attacks.*

Another bit of extra information enabled by OS detection is a guess at a target's uptime. This uses the TCP timestamp option (RFC 1323) to guess when a machine was last rebooted. The guess can be inaccurate due to the timestamp counter not being initialized to zero or the counter overflowing and wrapping around, so it is printed only in verbose mode

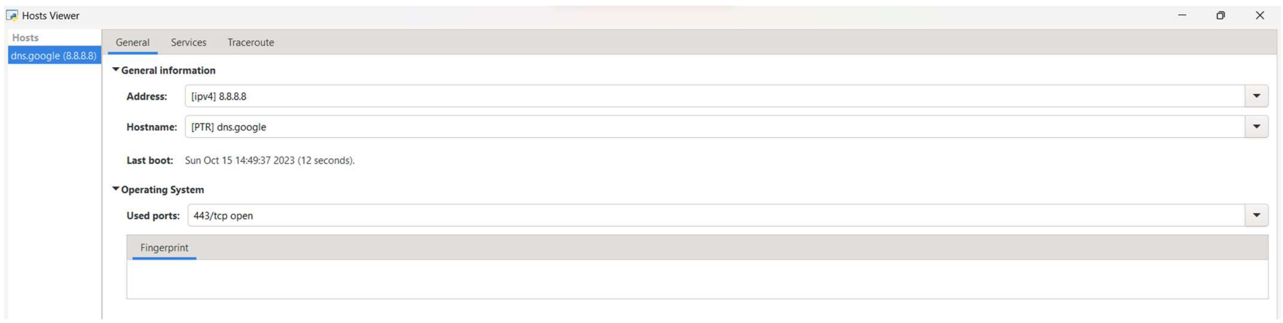


Target: 8.8.8.8 Profile: Intense scan

Command: nmap -T4 -A -v 8.8.8.8

nmap -T4 -A -v 17.112.92.0

```
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-15 14:50 India Standard Time
NSE: Loaded 156 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Initiating Ping Scan at 14:50
Scanning 17.112.92.0 [4 ports]
Completed Ping Scan at 14:50, 2.14s elapsed (1 total hosts)
Nmap scan report for 17.112.92.0 [host down]
NSE: Script Post-scanning.
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Initiating NSE at 14:50
Completed NSE at 14:50, 0.00s elapsed
Read data files from: C:\Program Files (x86)\Nmap
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.00 seconds
           Raw packets sent: 8 (304B) | Rcvd: 0 (0B)
```



General information

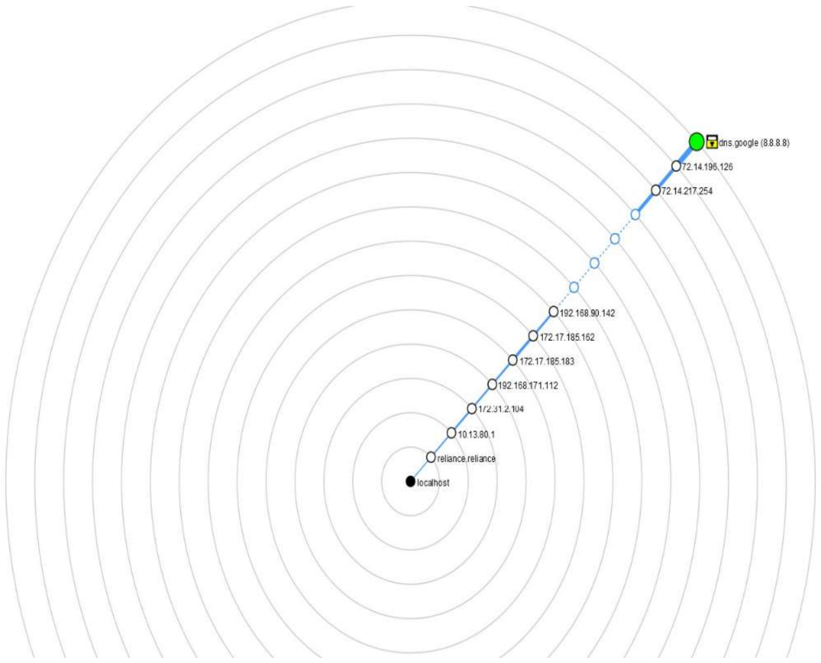
Address: [ipv4] 8.8.8.8

Hostname: [PTR] dns.google

Last boot: Sun Oct 15 14:49:37 2023 (12 seconds).

Operating System

Used ports: 443/tcp open


dns.google (8.8.8.8)
72.14.196.126
72.14.217.254
192.168.90.142
172.17.185.162
172.17.185.183
192.168.171.112
172.31.2.104
10.13.80.1
reliance.reliance
localhost

Exercise:16

Do the following using NS2 Simulator

i. NS2 Simulator-Introduction

ii. Simulate to Find the Number of Packets Dropped

iii. Simulate to Find the Number of Packets Dropped by TCP/UDP

iv. Simulate to Find the Number of Packets Dropped due to Congestion

v. Simulate to Compare Data Rate& Throughput.

i. NS2 Simulator-Introduction

NS2, or Network Simulator 2, is a widely used open-source network simulation tool that allows researchers and network professionals to simulate the behavior of computer networks. It's primarily used for studying the performance, behavior, and protocols of wired and wireless networks. NS2 is a discrete event simulator, meaning it models events occurring at specific points in time, such as data packet transmissions, arrivals, and departures.

Here's an introduction to NS2 and its key features:

*1. **Network Simulation:** NS2 is used to simulate a wide range of network types, including wired and wireless networks, mobile networks, ad-hoc networks, and more. Researchers and network engineers can use NS2 to test, evaluate, and analyze the performance of network protocols and algorithms.*

*2. **Event-Driven Simulation:** NS2 operates on the principle of discrete event simulation. It models network events as discrete points in time and simulates how these events affect the network. This approach provides high-level accuracy in modeling network behavior.*

*3. **Script-Based Configuration:** NS2 simulations are configured using scripts, most commonly written in Tcl (Tool Command Language). These scripts define the network topology, traffic patterns, routing protocols, and other simulation parameters. NS2*

scripts are highly customizable and allow you to design experiments tailored to your research needs.
4. **Customizability:** NS2 is highly customizable, which makes it a valuable tool for academic research and network protocol development. Researchers can modify existing protocols, implement new ones, and simulate various scenarios to test their behavior.
5. **Graphical Output:** NS2 provides graphical tools for visualizing simulation results, making it easier to understand and analyze the network's performance. You can generate various graphs and trace files to view statistics and network characteristics.
6. **Community Support:** NS2 has a large and active user community. This means you can find extensive documentation, tutorials, and support forums to help you get started and solve any issues you encounter.
7. **Extensive Library:** NS2 comes with an extensive library of network components and protocols, which simplifies the process of simulating a wide range of networking scenarios.
8. **Real-World Applications:** NS2 has been used in various research areas, such as computer networks, mobile and wireless communication, ad-hoc networks, sensor networks, and more. It's also used for developing and evaluating network protocols and algorithms.
9. **Legacy Software:** It's important to note that as of my last knowledge update in September 2021, NS2 had become somewhat of a legacy tool, with its successor, NS3 (Network Simulator 3), gaining popularity. NS3 is designed to be more modular and flexible. However, NS2 is still widely used and valuable for specific research and educational purposes.

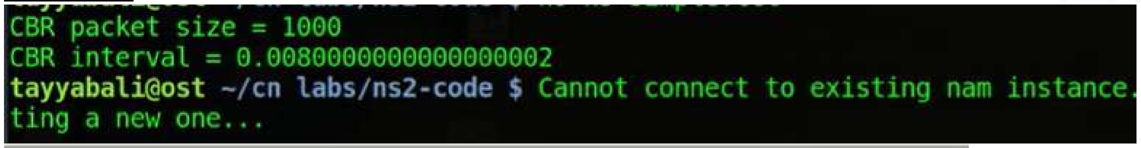
Program:

```
# Create a simulator object
set ns [new Simulator]

# Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

# Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

# Define a 'finish' procedure
proc finish {} {
   global ns nf
   $ns flush-trace
   # Close the NAM trace file
   close $nf
   # Execute NAM on the trace file
   exec nam out.nam &
   exit 0
}

# Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

# Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
```

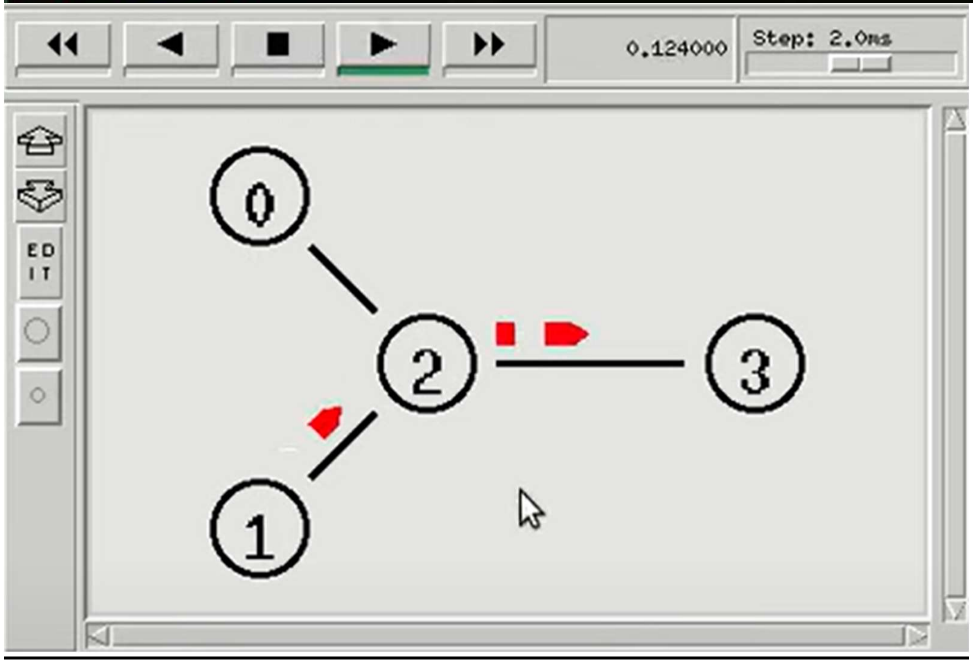
```
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

# Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

# Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

# Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

# Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

# Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP

# Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2
```

```
# Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false

# Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"

# Detach tcp and sink agents (not really necessary)
$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"

# Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

# Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

# Run the simulation
$ns run
```

Output:



```
CBR packet size = 1000
CBR interval = 0.0080000000000000002
tayyabali@ost ~/cn labs/ns2-code $ Cannot connect to existing nam instance.
ting a new one...
```


0.124000
Step: 2.0ms
ED
IT
0
2
3
1